# THE DATASHEET OF

# FMS7401LEN14

# FMS7401L

## Digital Power Controller

## General Description

The FMS7401L is a Digital Power Controller designed for applications requiring ease of digital based control over analog based implementations. The FMS7401L is an ideal solution to implement ballast control, motor control and battery management functions. It integrates a wide variety of analog blocks with an 8-bit microcontroller core to offer a complementary feature set with high performance, low power and small size in a single chip.

The FMS7401L is intended for applications using a supply voltage in the 2.7V to 3.6V range. It is fabricated using CMOS technology and is fully static offering a significant power savings. The FMS7401L is available in both 8-pin and 14-pin PDIP, SOIC and TSSOP packages.

## Features

- 8-bit Microcontroller Core
- 1K bytes on-board code EEPROM
- 64 bytes data EEPROM
- 64 bytes SRAM
- Watchdog Reset
- Multi-input Wakeup on all general purpose I/O pins
- Fast 12-bit PWM timer with dead time control and half-bridge output drive
  – Input Capture Mode
- 5-Ch 8-bit Analog-to-Digital Converter
  – 20 µS conversion time
  – Sample and Hold
  – Internal Voltage Reference (1.21V)
  – Gated Auto-sampling Mode
- Auto-zero Amplifier (gain 16)
- Uncommitted Amplifier
- Internal Current Source Generator (1mA)
- On-chip Oscillator
  – No external components
  – 1µs instruction cycle time
- On-chip Power-on Reset
- Programmable read and write disable functions
- Memory Mapped I/O
- Programmable Comparator (63 Levels)
- Brown-out Reset
- Software selectable I/O option
- Push-pull outputs with tri-state option
- Weak pull-up or high impedance inputs
- Fully static CMOS
  – Power Saving Halt Mode
    – < 1.3µA @ 3.3V
  – Power Saving Idle Mode
    – < 180µA @ 3.3V
- Single supply operation
  – 2.7V – 3.6V
- 40 years data retention
- 100,000 data changes
- 8-/14-pin PDIP, SOIC, and TSSOP packages
- In-circuit programming
  – Fast Page-write Programming Mode

| Device | Supply Voltage | Program Memory (bytes) | Data Memory (bytes) | | I/O | Pin Count |
|---|---|---|---|---|---|---|
| | | | SRAM | Data EEPROM | | |
| FMS7401L | 2.7V – 3.6V | 1K | 64 | 64 | 6 | 8 |
| FMS7401L | 2.7V – 3.6V | 1K | 64 | 64 | 8 | 14 |

# Block Diagram

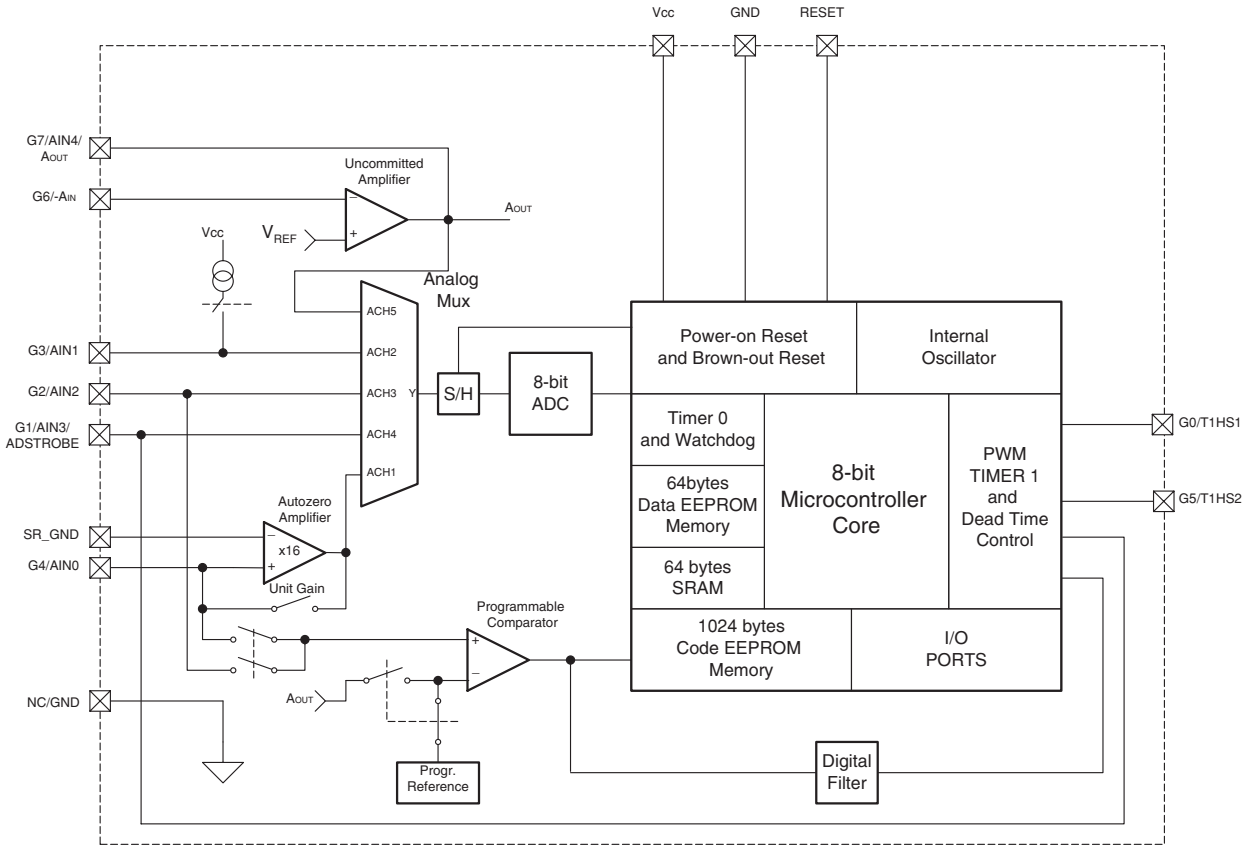
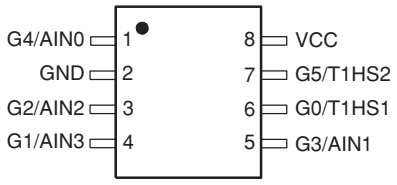
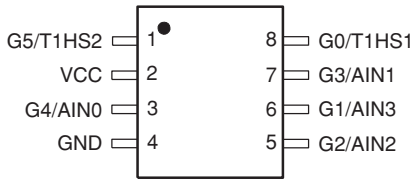



**Figure 1. FMS7401L Block and Connection Diagram**

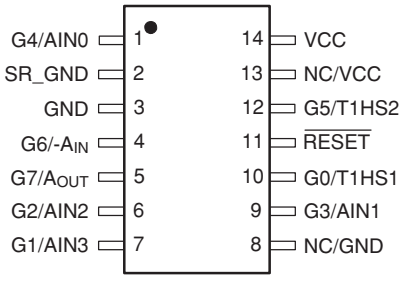
# Pin Configurations




**FMS7401L 8-Pin PDIP/SOIC**




**FMS7401L 8-Pin TSSOP**




**FMS7401L 14-Pin PDIP/SOIC/TSSOP**

# FMS7401L Pin Definitions

| Pin Number | | | Pin Name | Pin Function Description |
|---|---|---|---|---|
| 8-Pin | | 14-Pin | | |
| PDIP SOIC | TSSOP | PDIP SOIC TSSOP | | |
| 1 | 3 | 1 | G4/AIN0 | General purpose I/O port (bit 4 of the I/O configuration registers). |
| | | | | AIN0 analog input of the ADC (autozero amplifier's positive terminal). |
| | | | | Programmable Comparator non-inverting input, if COMPSEL=0. |
| 2 | 4 | 3 | GND | Digital ground pin. |
| 3 | 5 | 6 | G2/AIN2 | General purpose I/O port (bit 2 of the I/O configuration registers). |
| | | | | AIN2 analog input of the ADC. |
| | | | | Programmable Comparator non-inverting input, if COMPSEL=1. |
| 4 | 6 | 7 | G1/AIN3/ ADSTROBE | General purpose I/O port (bit 1 of the I/O configuration registers). |
| | | | | AIN3 analog input of the ADC. |
| | | | | External digital clock input. |
| | | | | PWM Timer 1's ADSTROBE output. |
| 5 | 7 | 9 | G3/AIN1 | General purpose I/O port (bit 3 of the I/O configuration registers). |
| | | | | AIN1 analog input of the ADC. |
| | | | | Internal current source generator pin. |
| 6 | 8 | 10 | G0/ T1HS1 | General purpose I/O port (bit 0 of the I/O configuration registers). |
| | | | | PWM Timer 1's T1HS1 output. |
| 7 | 1 | 12 | G5/ T1HS2 | General purpose I/O port (bit 5 of the I/O configuration registers). |
| | | | | PWM Timer 1's T1HS2 output. |
| 8 | 2 | 14 | VCC | Supply voltage input. |
| – | – | 2 | SR_GND | AIN0 analog input of the ADC (autozero amplifier's negative terminal). SR_GND is internally connected to GND in the 8-pin FMS7401L. |
| – | – | 4 | G6/-$A_{IN}$ | General purpose I/O port (bit 6 of the I/O configuration registers). |
| | | | | Uncommitted amplifier negative analog input. |
| – | – | 5 | G7/AIN4/ $A_{OUT}$ | General purpose I/O port (bit 7 of the I/O configuration registers). |
| | | | | AIN4 analog input of the ADC. |
| | | | | Uncommitted amplifier analog output. |
| – | – | 8 | NC/GND | In the FMS7401L, pin 8 is internally connected to GND. Externally, pin 8 should be left unconnected or connected to GND. |
| – | – | 11 | RESET | Active low external reset input. |
| – | – | 13 | NC/VCC | In the FMS7401L, VCC is internally connected to pin 13. Externally, pin 13 should either be left unconnected or connected to pin 13. |

# Table of Contents

**FMS7401L** . . . 1

General Description . . . 1
Features . . . 1
Block Diagram . . . 2
Pin Configurations . . . 2
FMS7401L Pin Definitions . . . 3

**1 Reset Circuit** . . . 8

1.1 Power-on Reset Circuit . . . 8
1.2 External Reset . . . 8
1.3 Brown-out Reset Circuit . . . 8

**2 Clock Circuit** . . . 10

2.1 PLL . . . 10

**3 Power Saving Modes** . . . 13

3.1 Halt Mode . . . 13
3.1.1 PLL Steps for Halt Mode . . . 13
3.2 Idle Mode . . . 14
3.2.1 PLL Steps for Idle Mode . . . 14

**4 ADC Circuit** . . . 16

4.1 ADC Circuit Configuration . . . 16
4.1.1 ADCNTRL1 Register . . . 17
4.1.2 ADCNTRL2 Register . . . 19
4.2 ADC Conversion Modes . . . 21
4.2.1 Analog Input Voltage and its 8-bit Digital Result . . . 22
4.2.2 ADC Gated Auto-sampling Mode . . . 22
4.2.3 ADC Conversion Clock Configuration . . . 22
4.3 Autozero Amplifier . . . 23
4.4 Uncommitted Amplifier . . . 23
4.5 Current Source Generator . . . 23

**5 Programmable Comparator Circuit** . . . 25

5.1 Programmable Comparator's Voltage Threshold Levels (VLOOP=0) . . . 25
5.2 Hardware Voltage and Current Loop Control (VLOOP=1) . . . 28
5.3 Digital Delay Filter with PWMOFF Output . . . 30

**6 PWM Timer 1 Circuit** . . . 32

6.1 PWM Timer 1 Configuration Registers . . . 32
6.1.1 PSCALE Register and Timer 1 Clock Configuration . . . 32
6.1.2 PWM Cycle Configuration Registers . . . 34
6.1.3 Timer 1 Control Register . . . 35
6.2 Pulse Width Modulation (PWM) Mode . . . 37
6.3 Input Capture Mode . . . 40

**7 Timer 0 Circuit** . . . 41

7.1 Idle Timer . . . 41
7.2 Watchdog Timer . . . 41

**8 I/O Ports** . . . 43

8.1 I/O Registers . . . 43

**9 Multi-input Wakeup Circuit** . . . 44

9.1 MIW Configuration Registers . . . 44

**10 8-Bit Microcontroller Core** . . . . . 46

10.1 Core Registers . . . . . 46
10.1.1 Accumulator (A) . . . . . 47
10.1.2 X-Pointer (X) . . . . . 47
10.1.3 Program Counter (PC) . . . . . 47
10.1.4 Stack Pointer (SP) . . . . . 47
10.1.5 Status Register (SR) . . . . . 47
10.1.6 Interrupt Handling . . . . . 49
10.2 Addressing Modes . . . . . 50

**11 Device Memory** . . . . . 54

11.1 Initialization Registers . . . . . 54
11.2 Memory Map . . . . . 57

**12 In-circuit Programming Specification** . . . . . 59

12.1 Programming Mode Interface . . . . . 59
12.2 Programming Protocol . . . . . 60
12.2.1 Byte Write Sequence . . . . . 60
12.2.2 Page Write Sequence . . . . . 60
12.2.3 Byte Read Sequence . . . . . 61
12.2.4 Program Memory Erase . . . . . 63

**13 Electrical Characteristics** . . . . . 64

13.1 FMS7401L (2.7V to 3.6V) . . . . . 65

**Ordering Information** . . . . . 77

**Physical Dimensions** . . . . . 78

# List of Figures

Figure 1. FMS7401L Block and Connection Diagram . . . . . 2
Figure 2. BOR and POR Circuit Relationship Diagram . . . . . 9
Figure 3. Internal Clock Scheme . . . . . 11
Figure 4. External Clock Scheme . . . . . 12
Figure 5. Recommended Halt/Idle Flow . . . . . 13
Figure 6. ADC Block Diagram . . . . . 17
Figure 7. Current Generator Interface . . . . . 24
Figure 8. Programmable Comparator Block Diagram (VLOOP = 0) . . . . . 26
Figure 9. Programmable Comparator Block Diagram (VLOOP = 1) . . . . . 29
Figure 10. Digital Delay Timing . . . . . 31
Figure 11. Timer 1's PWM Mode Block Diagram . . . . . 39
Figure 12. Example PWM Output Signals a) and b) . . . . . 39
Figure 13. Timer 1's Input Capture Mode Block Diagram . . . . . 40
Figure 14. PORTGD Logic Diagram . . . . . 43
Figure 15. Multi-input Wakeup (MIW) Block Diagram . . . . . 45
Figure 16. Core Program Model . . . . . 46
Figure 17. Basic Interrupt Structure . . . . . 49
Figure 18. Programming Mode Pin Configurations . . . . . 59
Figure 19. Programming Protocol . . . . . 62
Figure 20. Serial Data Timing . . . . . 62
Figure 21. Page Mode Protocol . . . . . 62
Figure 22. Internal Oscillator Frequency ($F_{OSC}$) vs. Temperature . . . . . 68
Figure 23. Icc Active vs. Temperature (no PLL or data EEPROM writes) . . . . . 68
Figure 24. Icc Active vs. Temperature (no PLL, with data EEPROM writes) . . . . . 69
Figure 25. Icc Active vs. Temperature (with PLL, no data EEPROM writes) . . . . . 69
Figure 26. Halt Current vs. Temperature . . . . . 70
Figure 27. Idle Current vs. Temperature (no PLL) . . . . . 70
Figure 28. Idle Current vs. Temperature (with PLL) . . . . . 71
Figure 29. $V_{OL}$ vs. $I_{OL}$ @ 25°C (G1–G4, G6, G7) . . . . . 71
Figure 30. $V_{OL}$ vs. $I_{OL}$ @ 25°C (G0, G5) . . . . . 72
Figure 31. $V_{OH}$ vs. $I_{OH}$ @ 25°C (G1–G4, G6, G7) . . . . . 72
Figure 32. $V_{OH}$ vs. $I_{OH}$ @ 25°C (G0, G5) . . . . . 73
Figure 33. BOR Level vs. Temperature . . . . . 73
Figure 34. Programmable Comparator Voltage Level vs. Temperature . . . . . 74
Figure 35. $V_{REF}$ vs. Temperature . . . . . 74
Figure 36. Current Source ($I_{SRC}$) vs. Temperature . . . . . 75
Figure 37. Gain 16 Error vs. Temperature . . . . . 75

# List of Tables


Table 1. Default Register States . . . . 8
Table 2. CMODE Bit Definition . . . . 10
Table 3. PLL Frequency Selection ($F_{PLL}/F_{OSC}$ = 2MHz) . . . . 10
Table 4. HALT Register Definition . . . . 13
Table 5. ADCNTRL1 Register Bit Definitions . . . . 19
Table 6. Analog Input Channel Selection (ACHSEL[3:0]) Bit Definitions . . . . 19
Table 7. ADCNTRL2 Register Bit Definitions . . . . 21
Table 8. Programmable Comparator (COMP) Control Register Bit Definitions . . . . 25
Table 9. Programmable Comparator Lower Voltage Reference $V_{THL}$ (Levels 1 – 31) . . . . 27
Table 10. Programmable Comparator Upper Voltage Reference $V_{THU}$ (Levels 32 – 63) . . . . 28
Table 11. Digital Delay (DDELAY) Register Bit Definitions . . . . 31
Table 12. Prescale (PSCALE) Register Bit Definitions . . . . 33
Table 13. PLL Divide Factor Selection Bits and the $F_{T1CLK}$ Resolution ($F_{OSC}$=2 MHz) . . . . 33
Table 14. Timer 1 Prescale Selection (PS) Bits . . . . 34
Table 15. Dead Time (DTIME) Register Bit Definitions . . . . 35
Table 16. Timer 1 Control (T1CNTRL) Register Bit Definitions . . . . 37
Table 17. Timer 1 Mode Configuration Bits . . . . 37
Table 18. Timer 0 Control (T0CNTRL) Register Definitions . . . . 41
Table 19. Watchdog Service Register (WDSVR) Definition . . . . 42
Table 20. I/O Register Bit Assignments . . . . 43
Table 21. I/O Configuration Options . . . . 43
Table 22. Multi-input Wakeup (MIW) Register Bit Assignments . . . . 45
Table 23. Interrupt Priority Sequence . . . . 48
Table 24. Instruction Addressing Modes . . . . 51
Table 25. Instruction Cycles and Bytes . . . . 52
Table 26. Initialization Register 1 Bit Definitions . . . . 55
Table 27. Initialization Register 3 Bit Definitions . . . . 55
Table 28. Initialization Register 4 Bit Definitions . . . . 55
Table 29. T1HS1 (G0) and T1HS2 (G5) Default Configuration . . . . 56
Table 30. Memory Mapped Registers . . . . 57
Table 31. Memory Mapped Registers and their Register Bit Definitions . . . . 58
Table 32. Programming Interface Electrical Characteristics . . . . 59
Table 33. 32-Bit Command and Response Word . . . . 61

# 1 Reset Circuit

The reset circuit in the FMS7401L contains four input conditions that trigger a main system reset. When the main system reset is triggered, a sequence of events occur defaulting all memory mapped registers (including the initialization registers) and I/Os to their initial states (see Table 1). During the system reset sequence, the instruction core execution is halted allowing time for the internal oscillator and other analog circuits to stabilize. Once the system reset sequence completes, the device will begin with its normal operation executing the instruction program residing in the code EEPROM memory. The time required for the system reset sequence to complete ($T_{RESET}$) is dependent on the individual trigger condition and is defined in the Electrical Characteristics section of the datasheet. The four reset trigger conditions are as follows:

- Power-on Reset (POR)
- External Reset[1]
- Brown-out Reset (BOR)
- Watchdog Reset[2]

**Table 1. Default Register States**

| Peripheral/Register | External Reset | POR |
|---|---|---|
| G1, G2, G3, G4, G6, G7 | High-impedance input (tri-state input) | |
| G0, G5 | Defined by Init Reg. 4 (see Table 28) | |
| SRAM Memory | No change | Unspecified |
| Stack Pointer | 0xF | 0xF |
| Status Register | 0x80 | 0x80 |
| T1CMPA, T1CMPB and T1RA Registers | 0xFFF | 0xFFF |
| DTIME Register | 0x1F | 0x1F |
| All other memory mapped register not listed above.[3] | 0x00 | 0x00 |

## 1.1 Power-on Reset Circuit

The Power-on Reset (POR) circuit maintains the device in a reset state until Vcc reaches a voltage level high enough to guarantee proper device operation. The POR circuit is sensitive to the different Vcc ramp rates and must be within $S_{Vcc}$ as specified in the Electrical Characteristics section of the datasheet.

The POR circuit does not generate a system reset when Vcc is falling. This feature is performed by the Brown-out Reset (BOR) circuit and must be enabled by the BOREN bit of the Initialization Register 1.[4] In the case where Vcc does not drop to 0V before the next power-up sequence, it is necessary to enable the BOR circuit and/or reset the device externally through the RESET pin.[1]

## 1.2 External Reset[1]

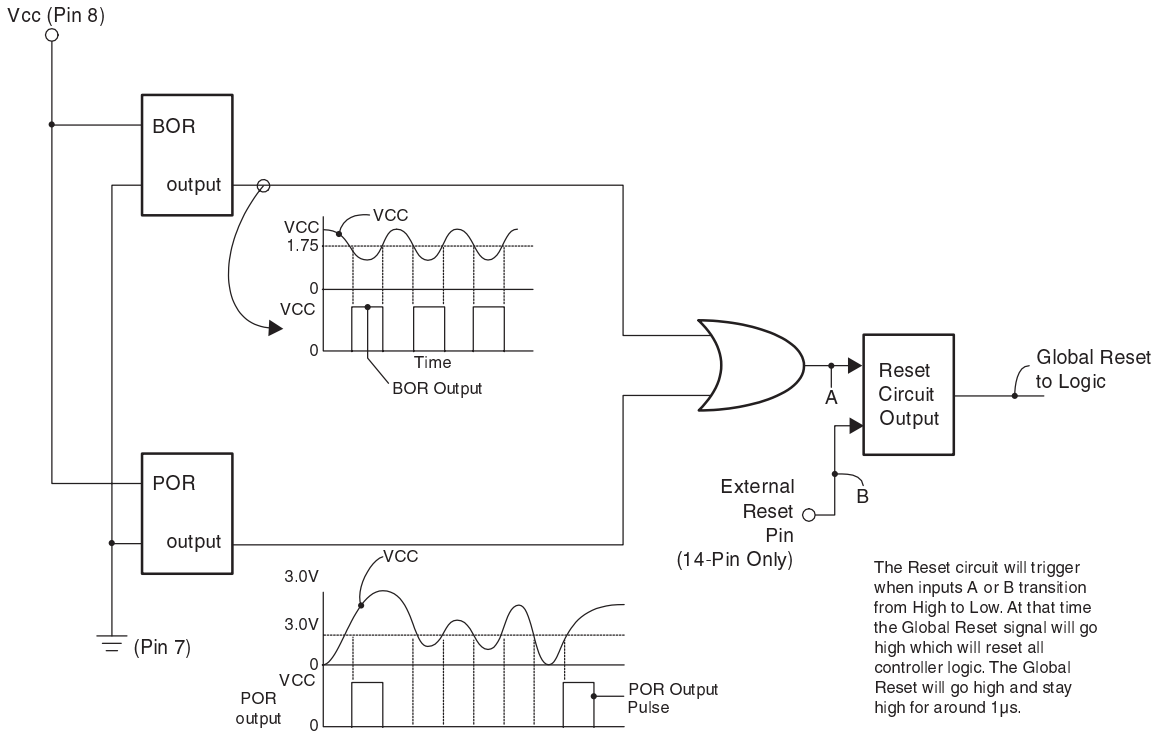
The device may be externally reset through the RESET input pin if the POR/BOR circuits cannot be used to properly reset the device in the application. The RESET input pin contains an internal pull-up resistor making it an active low signal. Therefore, to issue a device system reset the RESET input should be held low for at least 10µS before being released (i.e. returned to a high state). While the RESET input is held low, the internal oscillator and other analog circuits are kept in a low power state reducing the current consumption of the device (a state resembling Halt Mode). In addition, the I/O pins are all initialized to an input tri-state configuration unless defaulted otherwise.[5] At the rising edge of the RESET input signal, the main system reset sequence is triggered releasing the internal oscillator and other analog circuits so that they may be initialized and begin their normal operation.

## 1.3 Brown-out Reset Circuit

The Brown-out Reset (BOR) circuit is one of the on-chip analog comparator peripherals and must be enabled through the BOREN bit of the Initialization Registers 1.[4] The BOR circuit is used to hold the device in a reset state when Vcc drops below a fixed threshold defined in the Electrical Characteristics section of the datasheet. While in reset, the device is held in its initial condition until Vcc rises above the fixed/power-on threshold. Shortly after Vcc rises above the fixed/power-on threshold, the internal system reset sequence is started. Once the system reset sequence completes, the device will begin with its normal operation executing the instruction program residing in the code EEPROM memory.

The BOR circuit should be used in situations when Vcc rises and falls slowly and in situations when Vcc does not fall to 0V before rising back to the device's normal operating range. The BOR circuit can be thought of as a supplement function to the POR circuit if Vcc does not fall below 0.7V.

**Figure 2. BOR and POR Circuit Relationship Diagram**




1. Available only on the 14-pin package option.
2. Refer to the Timer 0 Circuit section of the datasheet for details regarding the Watchdog Reset.
3. Refer to Table 30 of the Device Memory section of the datasheet for the detailed memory map.
4. Refer to the Device Memory section of the datasheet for details regarding the Initialization Register 1.
5. Refer to Table 28 and Table 29 of the Device Memory section of the datasheet for details.

# 2 Clock Circuit

The FMS7401L may be clocked using its internal oscillator circuit or using an external digital clock signal. The desired clock source is selectable by the CMODE bit of the Initialization Register 1.[1] During the reset sequence, the CMODE bit is updated and the desired clock source (also called the device reference clock or $F_{RCLK1}$) takes control of the device. All devices are defaulted from the factory to use the internal oscillator as their main system instruction clock source. After power-up, the internal oscillator runs continuously unless entering Halt Mode or using an external clock source.

**Table 2. CMODE Bit Definition**

| CMODE | $F_{RCLK1}$ Clock Source |
|---|---|
| 0 | Internal Oscillator (@ $F_{OSC}$) |
| 1 | External digital clock (G1/AIN3) |

The internal oscillator signal is factory trimmed to yield the $F_{OSC}$ frequency as specified in the Electrical Characteristics section of the datasheet. If the external digital clock is selected, the input signal must have a 50/50 duty cycle, can range from DC to the $F_{OSC}$, and must be available upon power-up. When the device is driven using an external clock source, the clock input to the device should be provided through the AIN3/G1 input.

Once the source of $F_{RCLK1}$ is selected, the clock is then used as the reference clock for the PLL, the clock to the digital filter in the Programmable Comparator circuit, and is divided-by-2 to be used as the main system instruction clock ($F_{ICLK}$) of the device (see Figure 3 and Figure 4).

## 2.1 PLL

The FMS7401L has an internal digital clock multiplier (PLL) that steps-up the $F_{RCLK1}$ frequency by a multiplication factor of 32. The multiplied PLL output is then divided by a factor of 1, 2, 4, and 8 in order to generate its programmable output frequencies that may be used as the main system instruction clock or by the PWM Timer 1 circuit. The PLL provides the ability to run the PWM Timer 1 circuit at a frequency as high as 64MHz while the rest of the device operates at a much slower frequency keeping the total current consumption low.

The reference clock of the PLL is defined by the $F_{RCLK2}$ signal (as shown in Figure 3 and Figure 4) and sourced by the $F_{RCLK1}$ signal. In order to yield the proper output frequencies offered by the PLL, $F_{RCLK2}$ must operate at the $F_{PLL}$ frequency as specified in the Electrical Characteristics section of the datasheet. In the case that $F_{RCLK1}$ is operating at the upper $F_{OSC}$ frequency,[2] the REFBY2 bit in the ADCNTRL2 register[3] must be set in order to divide the $F_{RCLK1}$ by 2 to yield the appropriate $F_{PLL}$ frequency of the $F_{RCLK2}$ signal. Once the REFBY2 bit is set, the $F_{RCLK2}$ signal that drives the PLL and digital filter in the Programmable Comparator circuit operates at a $F_{RCLK1}/2$ frequency. If an external digital clock is sourcing $F_{RCLK1}$, the input signal must be supplied at the specified $F_{OSC}$ frequency in order to meet the specified $F_{PLL}$ frequency of the $F_{RCLK2}$ signal.

**Table 3. PLL Frequency Selection ($F_{PLL}/F_{OSC}$ = 2MHz)**

| FS[1:0] | | $F_{RCLK2}$ | $F_{ICLK}$ (FMODE = 0) | $F_{ICLK}$ (FMODE = 1) | $F_{PWMCLK}$ |
|---|---|---|---|---|---|
| 0 | 0 | 2 MHz | 1 MHz | 8 MHz | 8 MHz |
| 0 | 1 | 2 MHz | 1 MHz | 8 MHz | 16 MHz |
| 1 | 0 | 2 MHz | 1 MHz | 8 MHz | 32 MHz |
| 1 | 1 | 2 MHz | 1 MHz | 8 MHz | 64 MHz |

The PLL outputs may be used to clock both the PWM Timer 1 circuit and the main system clock. However, the PLL must first be enabled by setting the PLLEN bit of the PSCALE register.[4] Once set, the PLL is turned on and begins the locking phase. Before using any of the PLL outputs, software must wait the $T_{PLL_LOCK}$ to ensure that the PLL is locked into its appropriate frequency and in phase. The PLLEN bit may not be changed while the PWM Timer 1 circuit is in run mode.[5] Any write attempts to this bit during this condition will not change its value.

The PWM Timer 1 circuit may be clocked either by the PLL's $F_{PWMCLK}$ output or by the main system clock ($F_{ICLK}$). The FSEL bit of the PSCALE register[4] selects between the PLL's $F_{PWMCLK}$ output (if FSEL=1) or $F_{ICLK}$ (if FSEL=0). The FSEL bit may not be set if the PLL is not enabled (PLLEN=0) or changed while the PWM Timer 1 circuit is in run mode.[5] Any write attempts to this bit during these conditions will not change its value.

The FS[1:0] bits of the PSCALE register[4] select the divide factor for the $F_{PWMCLK}$ output (see Table 3). The FS bits may be changed by software at any time; however, if the PWM Timer 1 circuit is in run mode the FS[1:0] value will not change the $F_{PWMCLK}$ output frequency until after the PWM cycle ends (once the TMR1 counter overflows). The last FS[1:0] value at the PWM cycle end time will dictate the divide factor of the $F_{PWMCLK}$ output for the next PWM cycle. When reading the FS[1:0], the value reported will be the last value written by software (it may not necessarily reflect the divide factor for the current PWM cycle).

The main system instruction clock ($F_{ICLK}$) source may be provided by the internal oscillator ($F_{OSC}$) or the PLL's $F_{(FS=0)}$ output with the same divide factor as the FS[1:0] = 00 selection.[6] The FMODE bit of the PSCALE register[4] selects between the $F_{(FS=0)}$ (if FMODE=1) or $F_{RCLK1}$ divided-by-2 signal. With the FMODE bit enabled, it is possible to execute instructions at a speed eight times faster than the standard. The FMODE bit may not be set if the PLL is not enabled.[5] Any attempts to write to FMODE while PLLEN=0 will force FMODE=0 ignoring any set instruction. Once the PLL has been enabled, software may change $F_{ICLK}$'s source on-the-fly during normal instruction execution in order to speed-up a particular action.

In order to synchronously disable the PLL clocking structure, software must clear FSEL and FMODE before clearing the PLLEN bit in order to disable the PLL successfully e.g. using separate instructions like "RBIT PLLEN, PSCALE." There are also special conditions for Halt/Idle power saving modes that must also be considered. Please refer to the Power Saving Modes section of the datasheet for details.

**Figure 3. Internal Clock Scheme**

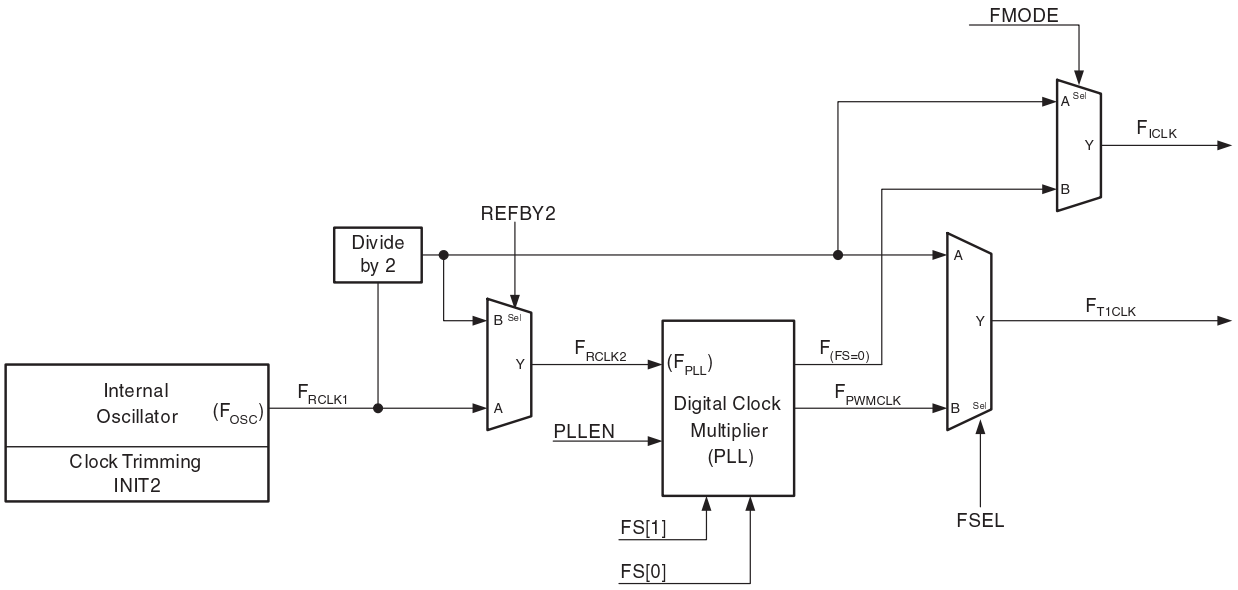
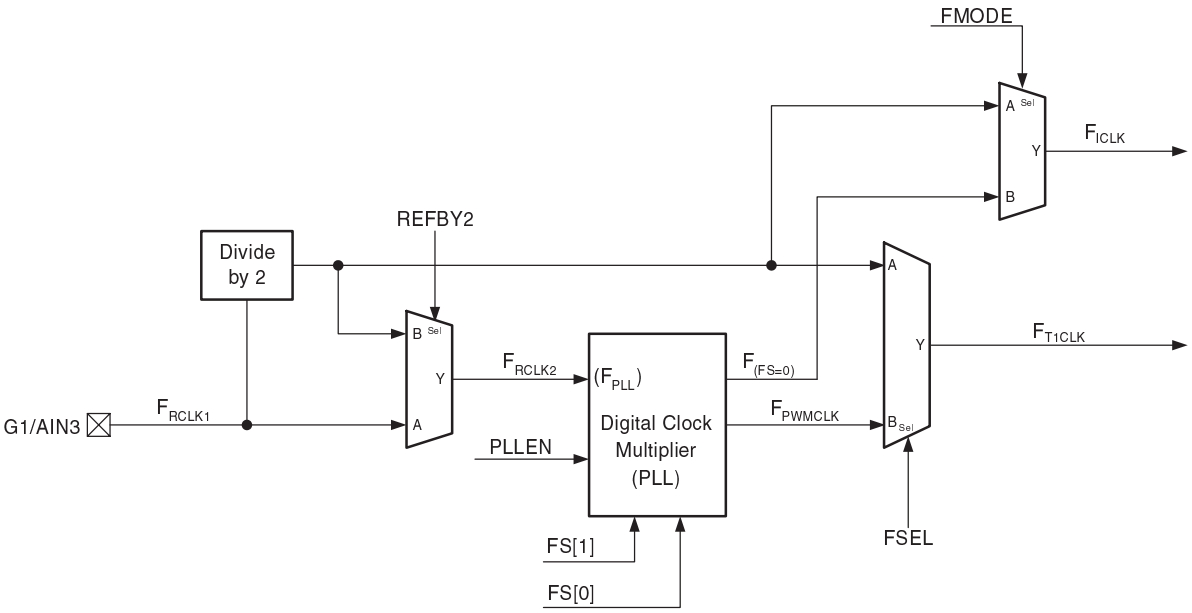


**Figure 4. External Clock Scheme**




1. Refer to the Device Memory section of the datasheet for details regarding the Initialization Registers 1.
2. The upper $F_{OSC}$ frequency (4MHz) is not a standard feature offered on the FMS7401L devices but is available upon request.
3. The ADCNTRL2 register is defined in the ADC Circuit section of the datasheet.
4. The PSCALE register is defined in the PWM Timer 1 Circuit section of the datasheet.
5. Software must always configure the device's entire clocking structure (see Figure 3 and Figure 4) while the PWM Timer 1 circuit is off (T1C0=0) and configured in PWM mode (T1C3=0).
6. The PLL's $F_{(FS=0)}$ output is not affected by the FS[1:0] bit value of the PSCALE register and merely shares the FS[1:0]=00 divide factor.

# 3 Power Saving Modes

The FMS7401L has both Halt and Idle power saving modes. Each mode is controlled by software and offers the advantage of reducing the total current consumption of the device in an application. For all current consumption details, please refer to the Electrical Characteristics section of the datasheet.

## 3.1 Halt Mode

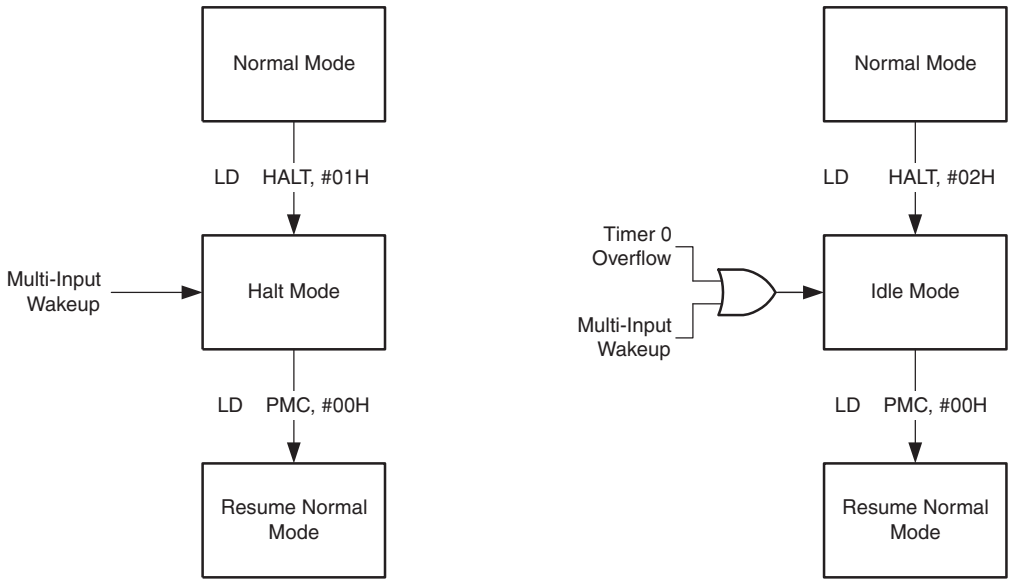
Halt Mode is a power saving feature that almost completely shuts down the device for current conservation. The device is placed into Halt Mode by setting the Halt enable bit (EHALT) of the HALT register using either the "LD M, #" or the "SBIT #, M" instructions in the software. EHALT is a write only bit and is automatically cleared upon exiting Halt Mode. When entering Halt Mode, the internal oscillator and all other on-chip systems including the Programmable Comparator (COMP) and Brown-out Reset (BOR) circuits are shut down.

The device can exit Halt Mode only by the Multi-input Wakeup (MIW) circuit.[1] Therefore, prior to entering Halt Mode, software must first configure the MIW circuit. After a wakeup from Halt Mode, a $T_{HALT_REC}$[2] start-up delay is initiated to allow the internal oscillator and other analog circuits to stabilize before normal device execution resumes. Immediately after exiting Halt Mode, software must clear the Power Mode Clear (PMC) register by using only the "LD M, #" instruction (see Figure 5).

**Table 4. HALT Register Definition**

| HALT Register (addr. 0xB7) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | EIDLE | EHALT |

**Figure 5. Recommended Halt/Idle Flow**




### 3.1.1 PLL Steps for Halt Mode

When using Halt Mode and the PLL in an application, software must take the appropriate steps in order to keep the integrity of the clock structure before entering and after exiting Halt since the PLL must be disabled. While in Halt Mode, all other device circuits except for the MIW are disabled. Once the PLL is disabled, all output frequencies are turned off. If the PLL is re-

enabled, it must complete the lock phase before software may enable the use of the outputs to clock any of the device circuits. Therefore, upon exiting Halt Mode software must wait the $T_{PLL_LOCK}$[2] to ensure that the PLL is locked into its appropriate frequency and in phase.

1. Initially, the PLLEN bit of the PSCALE register must be set in order to enable the PLL circuit.
2. If the PLL outputs are to be used to clock any of the device circuits, FMODE and/or FSEL of the PSCALE register must be set after the appropriate $T_{PLL_LOCK}$ wait time.[3]
3. Prior to entering Halt Mode, software must clear both FMODE and FSEL (the PWM Timer 1 must be disabled in order to clear either bit) keeping the PLLEN bit 1.
4. Using a separate instruction (e.g. RBIT PLLEN, PSCALE) disable the PLL by clearing the PLLEN bit.
5. Software may then instruct the device to enter Halt Mode.
6. If all disabled circuits must be re-enabled after exiting from Halt Mode, repeat all initial steps enabling all circuits in the appropriate order as well as waiting $T_{PLL_LOCK}$.

## 3.2 Idle Mode

In addition to the Halt Mode power saving feature, the device also supports an Idle Mode operation. The device is placed into Idle Mode by setting the Idle enable bit (EIDLE) of the HALT register through software using either the “LD M, #” or the “SBIT #, M” instructions. EIDLE is a write only bit and is automatically cleared upon exiting Idle Mode. The Idle Mode operation is similar to Halt Mode except the internal oscillator, PLL, and Timer 0 circuits remain active while the other on-chip systems including the Programmable Comparator (COMP) and Brown-out Reset (BOR) circuits are shut down.

The device exits Idle Mode automatically by the Timer 0 Idle overflow every 8192 cycles and by the Multi-input Wakeup (MIW) circuit.[1] Software must first configure the MIW prior to entering Idle Mode in order to wake the device from Idle without waiting for the overflow to occur. Once a wake from Idle Mode is triggered, the normal device execution resumes by the next clock cycle. Immediately after exiting Idle Mode, software must clear the Power Mode Clear (PMC) register by using only the “LD M, #” instruction (see Figure 5).

### 3.2.1 PLL Steps for Idle Mode

When using Idle Mode, the PLL does not need to be disabled prior to entering Idle as it does with Halt Mode. The PLL may remain enabled the entire time the device is in Idle; however, the device will consume additional current. If current consumption is important, consider using Halt instead of Idle Mode or at least disabling the PLL while in Idle.

By keeping the PLL enabled while in Idle Mode, the PLL’s outputs remain ready for use at any moment. With the PLL’s outputs available, software has the option to source the main system clock ($F_{ICLK}$) by the PLL’s $F_{(FS=0)}$ output when the FMODE bit of the PSCALE register is set.[3] In addition, if the PLL’s $F_{PWMCLK}$ output is clocking the PWM Timer 1 circuit,[4] the timer may remain operational while in Idle Mode. However, the total current consumption will increase, hence the recommendation to disable the PWM Timer 1 before entering Idle Mode. In contrast, if $F_{ICLK}$ is clocking the PWM Timer 1, the timer circuit execution (like the main system controller) is stopped during Idle. Whether the PWM Timer 1 is operational or not during Idle Mode, the instruction execution is stopped therefore all pending flags, etc. cannot be serviced.

If the PLL is to be disabled prior to entering Idle Mode, software must take the appropriate steps in order to keep the integrity of the clock structure. Once the PLL is disabled, all output frequencies are turned off. If the PLL is re-enabled, it must complete the lock phase before software may enable the use of the outputs to clock any of the device circuits. Therefore, upon exiting Idle Mode software must wait the $T_{PLL_LOCK}$ to ensure that the PLL is locked into its appropriate frequency and in phase.

1. Initially, the PLLEN bit of the PSCALE register must be set in order to enable the PLL circuit.
2. If the PLL outputs are to be used to clock any of the device circuits, FMODE and/or FSEL of the PSCALE register must be set after the appropriate $T_{PLL_LOCK}$ wait time.
3. Prior to entering Idle Mode, software must clear both FMODE and FSEL (the timer must be disabled in order to clear either bit) keeping the PLLEN bit 1.
4. Using a separate instruction (e.g. RBIT PLLEN, PSCALE) disable the PLL by clearing the PLLEN bit.
5. Software may then instruct the device to enter Idle Mode.
6. If all disabled circuits must be re-enabled after exiting from Idle Mode, repeat all initial steps enabling all circuits in the appropriate order as well as waiting $T_{PLL_LOCK}$.

---

1. The MIW and Timer 0 Circuits are described later in the datasheet.
2. Refer to the Electrical Characteristics section of the datasheet for details.
3. Refer to the Clock Circuit's PLL section of the datasheet for details.
4. The FSEL bit in the PSCALE register must be set.

# 4 ADC Circuit

The Analog-to-Digital Converter (ADC) Circuit extends the features of the FMS7401L by offering a 5-channel 8-bit ADC. The ADC may be programmed to convert voltages on any of the eight inputs of the analog mux, where five are multifunction input channels (ACH1-ACH5) and three are used for system calibration. The integrated ADC function offers a single cost-effective solution for applications requiring voltage, current and temperature sensing. The multifunction input channels may be configured to perform standard conversions on any of the analog input pins (G4/AIN0, G3/AIN1, G2/AIN2, G3/AIN3 or G7/AIN4). Three of the multifunction input channels may be programmed to perform ADC conversions through the internal Autozero Amplifier, Uncommitted Amplifier, and Current Source Generator for special control system and battery management applications (see Figure 6).

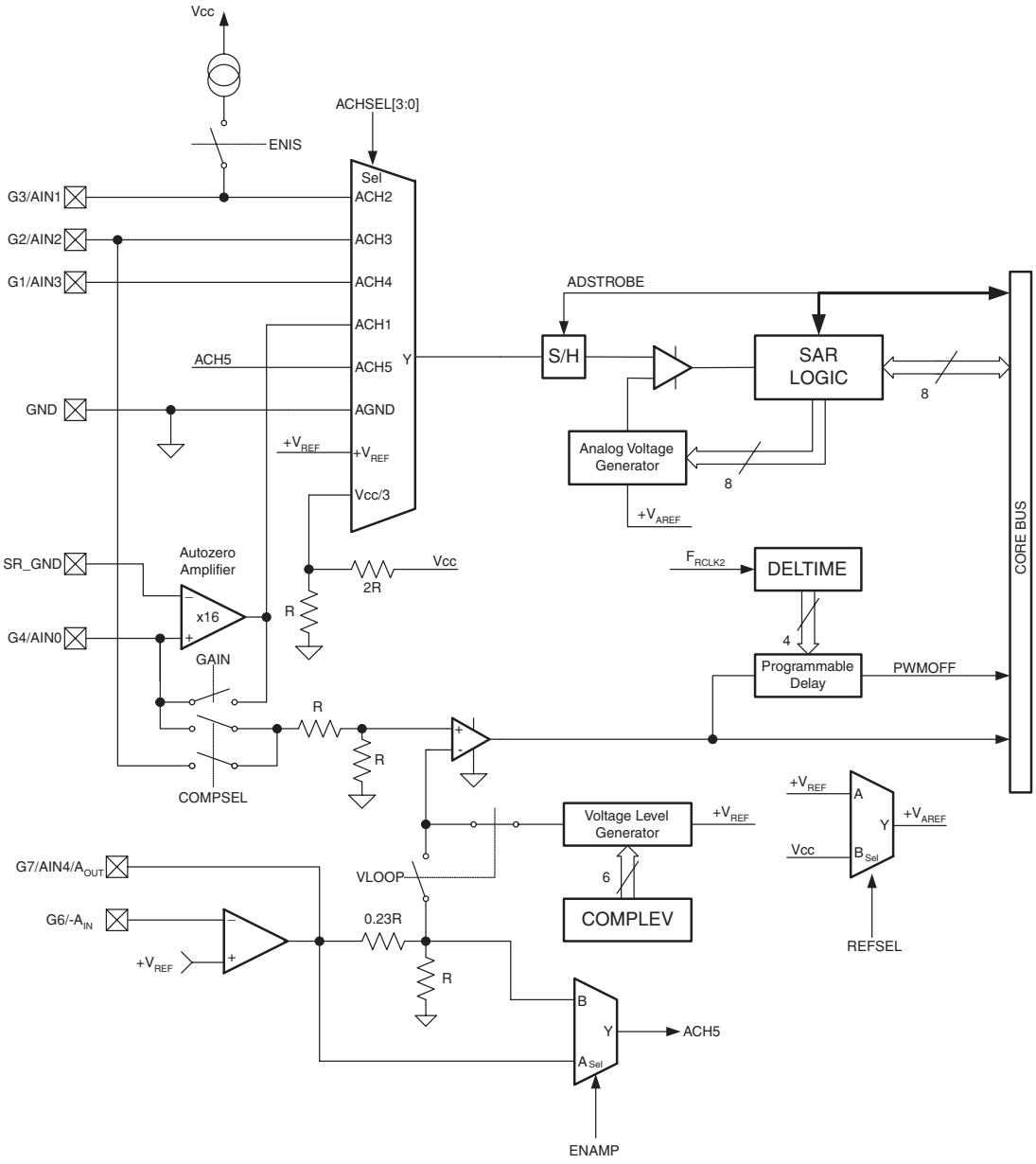
The ADC Circuit's eight analog inputs are software selectable where their analog input voltage is converted with respect to the internal ADC reference voltage ($V_{AREF}$). $V_{AREF}$ may be programmed to use the internal bandgap reference voltage ($V_{REF}$) or Vcc as its source. By default, the ADC circuit's $V_{AREF}$ is configured to use the internal $V_{REF}$ as its source.[1]

The ADC performs conversions of 8-bit resolution with accuracy as defined in the Electrical Characteristics section of the datasheet. For a standard ADC conversion, the ADC circuit converts the analog input voltage in a total of 13 conversion clock cycles, and a total of 20 conversion clock cycles when performing an autozero ADC conversion. To yield a better ADC conversion accuracy, the ADC circuit may configure the ADC clock ($F_{ADCLK}$) to a slower frequency, lengthening the total conversion time while improving its accuracy. As part of the total conversion time, the ADC circuit completes a sample and hold phase to measure fast changing analog signals before converting the voltage. An ADC conversion can be initiated by a software command or automatically (using the gated auto-sampling mode) by the active (on) edge transition of the ADSTROBE PWM Timer 1 output.[2] If enabled, the ADC circuit offers the use of its microcontroller hardware interrupt (ADCI) triggered after each completed ADC conversion so that the microcontroller core is freed to perform other tasks.

## 4.1 ADC Circuit Configuration

Software must access the three memory mapped ADC registers to configure and control the ADC circuit.[3] The ADC Control 1 (ADCNTRL1) register is used to select the analog input channel and ADC reference voltage ($V_{AREF}$) for the conversion. In addition, it is used to initiate a conversion through software, monitor the ADC pending flag, and enable the ADC circuit's microcontroller hardware interrupt (ADCI). The ADC Control 2 (ADCNTRL2) register is used to enable the internal Autozero Amplifier, Uncommitted Amplifier, Current Source Generator, and/or ADC Auto-sampling Mode. The ADCNTRL2 register is also used to divide the ADC $F_{ADCLK}$ clock to improve the conversion accuracy. Lastly, the ADC Data (ADATA) register is used by software to read the final converted 8-bit digital value. ADATA is a read only register and is updated automatically at the end of each ADC conversion.

**Figure 6. ADC Block Diagram**[4]




### 4.1.1 ADCNTRL1 Register

The ADCNTRL1 is an 8-bit memory map register used to configure and control the ADC circuits. Software has both read and write access to all bits of the register.

Bit 7 of the ADCNTRL1 register is the ADC pending (APND) flag and is triggered after the 8-bit converted digital value is latched to the ADATA register towards the end of the ADC conversion cycle. The APND bit may be used by software to monitor when to access ADATA or to issue microcontroller hardware interrupts (if enabled). In order for software to monitor APND, it must be cleared before the next converted value is latched in ADATA where the APND flag is set to 1.

Bit 6 of the ADCNTRL1 register is the ADC's microcontroller hardware interrupt enabled (AINTEN) bit. If set, hardware interrupts (ADCI) are enabled and triggered by the APND pending flag.[5] As long as the ADC pending flag is set, the hardware interrupt will continue to execute software's ADC interrupt service routine until the pending flag is cleared.[6]

Bit 5 of the ADCNTRL1 register is the ADC Conversion Start/Busy (ASTART) bit. Software must set the ASTART bit to initiate an ADC conversion when the ENDAS bit of the ADCNTRL2 register is set to 0. The ASTART bit will remain high as long as an ADC conversion is in progress (whether software or the ADSTROBE signal triggered the conversion). If software attempts to clear the ASTART bit while a conversion is in progress, the write command is ignored and the ASTART bit remains high until the conversion cycle completes. Software should monitor ASTART to determine when the conversion has completed instead of the APND bit. The APND bit may be triggered before the ASTART is automatically cleared. The ADC conversion completion delay may occur when the $F_{ICLK}$ clock is slower than an ADC conversion clock cycle.

Bit 4 of the ADCNTRL1 register is the ADC Voltage Reference Selection (REFSEL) bit. If REFSEL=0, the ADC Reference Voltage ($V_{AREF}$) becomes sourced by the internal bandgap voltage reference ($V_{REF}$). If REFSEL=1, the ADC Reference Voltage ($V_{AREF}$) becomes sourced by Vcc. If the ADC circuit is performing a conversion, software must avoid writing to the REFSEL bit.

Bits 3-0 of the ADCNTRL1 register are the Analog Channel Selection (ACHSEL[3:0]) bits selecting one of the eight analog input channels to convert its voltage (see Table 6). Software may write to the ACHSEL bits at any time; however, the actual ACHSEL selection signals will not change while an ADC conversion is in progress. If a read command is issued while a conversion is in progress, the current value of the ACHSEL bits may not necessarily reflect the actual state of the ACHSEL selection signals. The last value of the ACHSEL bits written by software at the time of the ADC conversion trigger, dictates the state of the ACHSEL selection signals for the triggered ADC conversion cycle.

The SBIT or RBIT instructions may be used to either set or clear one of the ADCNTRL1 register bits, like the AINTEN bit. The SBIT and RBIT instructions both take two instruction clock cycles to complete their execution. In the first cycle, all register bits are automatically read to obtain their most current value. In the second cycle, the bit to be set/cleared is given its new value and all bits are then re-written to the register. Using the SBIT/RBIT instruction to set/clear an enable bit with a pending flag in the same register may cause a potential hazard. Software may inadvertently clear a recently triggered pending flag if the trigger happened during the second phase of the SBIT/RBIT instruction execution. To avoid this condition, the LD instruction must be used to set or clear the interrupt enable bit. The ADC circuit is designed such that software may not trigger a pending flag by writing a 1 to the APND bit, it may only be cleared. The action of writing a 1 to the APND register bit holds its current bit value. The action of writing a 0 to the APND register bit clears the bit value. Therefore, the "LD T1CNTRL, #0E0H" instruction will set the ASTART and AINTEN bits without clearing APND.

**Table 5. ADCNTRL1 Register Bit Definitions**

| ADCNTRL1 Register (addr. 0x9F) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| APND | AINTEN | ASTART | REFSEL | ACHSEL[3:0] | | | |

| Bit | Description |
|---|---|
| APND | (0) ADC's pending flag is cleared. |
| | (1) ADC's pending flag is triggered. |
| AINTEN | (0) Disables ADC hardware interrupts. |
| | (1) Enables ADC hardware interrupts. |
| ASTART | (0) ADC conversion is not in progress. |
| | (1) Start an ADC conversion / ADC conversion in progress. |
| REFSEL | (0) ADC Reference ($V_{AREF}$) = Internal $V_{REF}$ |
| | (1) ADC Reference ($V_{AREF}$) = Vcc |
| ACHSEL[3:0] | Analog Input Channel Selection Bits. Refer to Table 6 for details. |

**Table 6. Analog Input Channel Selection (ACHSEL[3:0]) Bit Definitions**

| ACHSEL[3] | ACHSEL[2] | ACHSEL[1] | ACHSEL[0] | Analog Channel | I/O Equiv. |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | ACH1 | G4/AIN0 |
| 0 | 0 | 0 | 1 | ACH2 | G3/AIN1 |
| 0 | 0 | 1 | 0 | ACH3 | G2/AIN2 |
| 0 | 0 | 1 | 1 | ACH4 | G1/AIN3 |
| 1 | 0 | 0 | 0 | ACH5 | G7/AIN4/$A_{OUT}$ |
| 1 | 0 | 0 | 1 | GND | - |
| 1 | 0 | 1 | 0 | $+V_{REF}$ | - |
| 1 | 1 | 0 | 0 | Vcc/3 | - |

### 4.1.2 ADCNTRL2 Register

The ADCNTRL2 is an 8-bit memory map register used to configure the analog circuits. Six of the eight register bits are used to configure circuits directly related to the ADC circuit while the others are not related.

Bit 7 (REFBY2) of the ADCNTRL2 register is the reference clock ($F_{RCLK1}$) divide-by-2 enable bit. The REFBY2 bit configures the reference clock of the PLL and Programmable Comparator circuit to be sourced either by $F_{RCLK1}$ or $F_{RCLK1}/2$ clock. Refer to the Clock Circuit section of the datasheet for additional details.

Bit 6 (COMPSEL) of the ADCNTRL2 register is the Programmable Comparator's non-inverting input selection bit. If COMPSEL=0, the non-inverting input of the Programmable Comparator is the G4/AIN0 device pin. If COMPSEL=1, the non-inverting input of the Programmable Comparator is the G2/AIN2 device pin. Before enabling the Programmable Comparator circuit, the selected analog input port pin must be configured as a tri-state input bypassing the I/O circuitry.[9] Refer to the Programmable Comparator Circuit section of the datasheet for addition details.

Bit 5 of the ADCNTRL2 register is the Uncommitted Amplifier Enable (ENAMP) bit. If ENAMP=0, the Uncommitted Amplifier circuit is disabled and its pin connections (G6/-$A_{IN}$ and G7/$A_{OUT}$) may be used as normal I/O ports. The G7/AIN4 pin may still be used as a standard ADC conversion input through the analog ACH5 channel. If ENAMP=1, the Uncommitted Amplifier circuit is enabled and its pin connections must be configured as tri-state inputs where G6/-$A_{IN}$ is the inverting input and G7/$A_{OUT}$ is the amplifier output.[9] If the ADC circuit is performing a conversion on the analog ACH5 input when driven by the Uncommitted Amplifier, software must avoid clearing the ENAMP bit. Refer to the following Uncommitted Amplifier section for additional details.

Bit 4 (ENDAS) of the ADCNTRL2 register enables the ADC conversion's gated auto-sampling operating mode. If ENDAS=1, the ADC circuit configures the $F_{ADCLK}$ clock for synchronization with the PWM Timer 1's ADSTROBE output signal. The ADC circuit will then accept triggers by the active (on) edge transition of the ADSTROBE signal. All other ADC configuration

options must be prepared prior to setting the ENDAS bit. Refer to the following ADC Gated Auto-sampling Mode section for additional details. The ADSTROBE signal is generated by the PWM Timer 1 circuit and is configured using its T1CMPB and T1RA registers. Refer to the PWM Timer 1 Circuit section of the datasheet for details regarding its operation. If ENDAS=0, the ADC circuit is configured to accept only ADC start commands issued by software when setting the ASTART bit of the ADCNTRL1 register to 1. Refer to the following ADC Conversion Modes section for additional details.

Bits 3 and 2 (ASPEED[1:0]) of the ADCNTRL2 register selects the divide factor (1, 2, 4, or 8) to slow the $F_{ADCLK}$ clock extending the ADC conversion cycle time. In most cases, the $F_{ADCLK}$ clock division is performed to improve the ADC conversion accuracy. Refer to the following ADC Conversion Clock Configuration section for addition details.

Bit 1 of the ADCNTRL2 register is the Current Source Generator Enable (ENIS) bit. If ENIS=0, the Current Source Generator circuit is disabled and its G3/AIN1 pin may be used as a normal I/O port or as a standard ADC conversion input through the analog ACH2 channel. If ENIS=1, the Current Source Generator circuit is enabled and its pin connection must be configured as a tri-state input bypassing the I/O circuitry.[9] If the ADC circuit is performing a conversion on the analog ACH2 input when driven by the Current Source Generator, software must avoid clearing the ENIS bit. Refer to the following Current Source Generator section for additional details.

Bit 0 (GAIN) of the ADCNTRL2 register is the autozero amplifier enable bit. If GAIN=0, the autozero amplifier with its gain 16 circuitry is disabled where its G4/AIN0 pin connections may be used as a normal I/O port. The G4/AIN0 pin may still be used as a standard ADC conversion input through the analog ACH1 channel. If GAIN =1, the autozero amplifier with its gain 16 circuitry is enabled and its G4/AIN0 pin connection must be configured as a tri-state input where G4/AIN0 is the non-inverting and SR_GND is the inverting input of the amplifier.[9] Software may write to the GAIN bit at any time; however, the actual GAIN enable signal will not change while an ADC conversion is in progress. If a read command is issued while a conversion is in progress, the current value of the GAIN bit may not necessarily reflect the actual state of the GAIN enable signal. The last value of the GAIN bit written by software at the time of the ADC conversion trigger, dictates the state of the GAIN enable signal for the triggered ADC conversion cycle. Refer to the following Autozero Amplifier section for additional details.

**Table 7. ADCNTRL2 Register Bit Definitions**

| ADCNTRL2 Register (addr. 0xA0) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| REFBY2 | COMPSEL | ENAMP | ENDAS | ASPEED[1:0] | | ENIS | GAIN |

| Bit | Description |
|---|---|
| REFBY2 | Used to divide the reference clock for the PLL and digital filter of the Programmable Comparator circuit. Refer to the Clock Circuit section of the datasheet for details.<br>(0) $F_{RCLK2}=F_{RCLK1}$<br>(1) $F_{RCLK2}=F_{RCLK1}/2$ |
| COMPSEL | (0) G4 is connected to the Programmable Comparator's non-inverting input.<br>(1) G2 is connected to the Programmable Comparator's non-inverting input. |
| ENAMP | (0) Disables the Uncommitted Amplifier (G6 and G7 are normal I/Os).<br>(1) Enables the Uncommitted Amplifier where G6/-$A_{IN}$ is the inverting input and G7/$A_{OUT}$ is the amplifier output. The amplifier output ($A_{OUT}$) is also the ACH5 input to the ADC's analog mux. |
| ENDAS | (0) Enables the Standard ADC Conversion Mode where software must trigger an ADC conversion by setting the ASTART bit of the ADCNTRL1 register.<br>(1) Enables the ADC Gated Auto-sampling Mode where PWM Timer 1's ADSTROBE output to automatically triggers an ADC conversion. |
| ASPEED[1:0] | (0) ADC conversion clock speed = $F_{ADCLK}$<br>(1) ADC conversion clock speed = $F_{ADCLK}/2$<br>(2) ADC conversion clock speed = $F_{ADCLK}/4$<br>(3) ADC conversion clock speed = $F_{ADCLK}/8$ |
| ENIS | (0) Disable Current Source Generator (G3 is a normal I/O).<br>(1) Enable Current Source Generator where G3/AIN1 sources the $I_{SRC}$. |
| GAIN | (0) Disables the Autozero Amplifier (G4 is a normal I/O).<br>(1) Enables the Autozero Amplifier (with a gain of 16) where G4/AIN0 is its non-inverting input and SR_GND is it inverting input. |

## 4.2 ADC Conversion Modes

The ADC circuit may be configured to convert analog voltages with a conversion cycle time determined by the ADC clock ($F_{ADCLK}$) and the ASPEED bits of the ADCNTRL2 register. Refer to the following ADC Conversion Clock Configuration section for details. By default, the ADC circuit performs a conversion with every trigger initiated by software setting the ASTART bit of the ADCNTRL1 register to 1. The ADC circuit may also be configured to perform a conversion automatically (using the gated auto-sampling mode) with every active (on) edge of the PWM Timer 1 ADSTROBE output signal. Refer to the following ADC Gated Auto-sampling Mode section for details.

Before any ADC conversion triggers are issued (by software or automatically) software must configure the voltage reference ($V_{AREF}$) and analog input channel appropriately. This is done by programming the REFSEL and ACHSEL bits of the ADCNTRL1 register. If using the internal Autozero Amplifier, Uncommitted Amplifier, and Current Source Generator circuits in the application, software must also configure and enable the desired circuits. Lastly, the $F_{ADCLK}$ must be configured to improve the ADC conversion accuracy.

When performing an ADC conversion where software triggers the conversion, the ASTART bit of the ADCNTRL1 register remains high (1) symbolizing that a conversion is in progress. The ADC conversion is divided in two phases lasting a total of 13 conversion clock cycles. However, an autozero ADC conversion lasts a total of 20 conversion clock cycles. Refer to the following Autozero Amplifier section for details. In the first phase of a standard conversion, occupying the first four conversion cycles, the ADC circuit performs a sample and hold operation to measure fast changing analog signals before converting the input voltage. The second phase, occupying the last nine cycles, converts the analog input voltage to an 8-bit digital value and stores it in the ADATA register for easy access by software. Once the converted value is stored in ADATA, the APND bit is triggered and ASTART bit is cleared (symbolizing completion of the conversion cycle). Software cannot rely on the APND bit for

this information because the APND bit may be triggered before the ASTART is automatically cleared. The ADC conversion completion delay may occur when the $F_{ICLK}$ clock is slower than an ADC conversion clock cycle.

### 4.2.1 Analog Input Voltage and its 8-bit Digital Result

The relationship between the 8-bit digital value stored in the ADATA register and the analog input voltage is as follows:

$$V_{ADC} = \frac{V_{ACH(x)}}{V_{AREF}} \times 255$$

- $V_{ADC}$ is the 8-bit digital result of an ADC conversion.
- $V_{ACH(x)}$ is the analog voltage applied to the selected input channel.

### 4.2.2 ADC Gated Auto-sampling Mode

The ADC circuit may be configured in Gated Auto-sampling Mode by setting the ENDAS bit of the ADCNTRL2 register. When in Auto-sampling Mode, all ADC conversions are automatically triggered by the active (on) edge transition of the PWM Timer 1's ADSTROBE output signal.[2] If the period of the PWM ADSTROBE signal is less than the total ADC conversion time, any triggers issued while a conversion is in progress (ASTART=1) are ignored. Once the trigger is detected, the ASTART bit of the ADCNTRL1 register is set symbolizing that a conversion is in progress. The initial conversion phase, the sample and hold or autozero (if GAIN=1), begins after a 1µS cycle delay.[7] Once all eight digital bits are determined and stored in the ADATA register, the APND flag is set to trigger a hardware interrupt (if enabled) flagging software that the ADATA register has been updated with the ADC conversion results. Once all phases of the ADC conversion cycle completes, the ASTART bit is then automatically cleared by the ADC circuit. Since software cannot change the ADC circuit configuration while an ADC conversion is in progress, the ASTART bit must be monitored to determine when the conversion cycle completes. Software cannot rely on the APND bit for this information because the APND bit may be triggered before the ASTART is automatically cleared. The ADC conversion completion delay may occur when the $F_{ICLK}$ clock is slower than an ADC conversion clock cycle.

### 4.2.3 ADC Conversion Clock Configuration

The ADC conversion clock ($F_{ADCLK}$) is sourced either by the device's main system instruction clock ($F_{ICLK}$) or the PWM Timer 1's clock ($F_{T1CLK}$) depending on the ADC circuit's operating mode. If the standard ADC conversion mode is selected, the ADC circuit is automatically configured to source the $F_{ADCLK}$ clock by the $F_{ICLK}$ clock. If the ADC Conversion Auto-sampling Mode is selected, the ADC circuit is automatically configured to source the $F_{ADCLK}$ clock by the $F_{T1CLK}$ clock to synchronize the ADC conversions with the active (on) edge of the PWM Timer 1 ADSTROBE output signal.[2]

When in standard ADC conversion mode, the ASPEED[1:0] bits of the ADCNTRL2 register may be used to slow the total conversion time improving the ADC conversion accuracy. However, if the $F_{ICLK}$ clock is sourced by the PLL's $F_{(FS=0)}$ output (when FMODE=1) the $F_{ADCLK}$ will clock eight times faster than the proper conversion rate (1µS cycle time). The $F_{ADCLK}$ clock must then be divided by setting the ASPEED[1:0]=3 divide factor to yield a $F_{ADCLK}/8$ conversion clock cycle. Otherwise, software may temporarily clear FMODE returning the conversion cycle to its proper frequency and free the ASPEED bits to be used to improve the conversion accuracy. In addition, if the internal oscillator is trimmed to its upper $F_{OSC}$ frequency and it is sourcing the $F_{ICLK}$ clock, the ASPEED[1:0]=1 divided factor must be selected to yield a $F_{ADCLK}/2$ conversion clock cycle.[8] A greater divide factor may still be selected by setting the ASPEED[1:0]>1.

When in ADC Conversion Auto-sampling Mode, the ADC circuit automatically configures the $F_{ADCLK}$ clock to be sourced by the $F_{T1CLK}$ clock so that the ADC conversions may be synchronized with the active (on) edge of the ADSTROBE signal. However, the $F_{T1CLK}$ clock is first sent into a special divide circuit which evaluates its configuration to determine the divide factor needed to yield the proper $F_{ADCLK}$ conversion rate (1µS cycle time). The FMODE, FSEL, and FS bits of the PSCALE register are evaluated so that the divide circuit applies the appropriate divide factor to the $F_{T1CLK}$ clock (the PS bits do not apply). The ASPEED[1:0] bits of the ADCNTRL2 register may be used to slow the total conversion time improving the ADC conversion

accuracy. However, if the internal oscillator is trimmed to its upper $F_{OSC}$ frequency while FMODE and FSEL are zero, the ASPEED[1:0]=1 divided factor must be selected to yield a $F_{ADCLK}/2$ conversion clock cycle.[8] A greater divide factor may still be selected by setting the ASPEED[1:0]>1.

## 4.3 Autozero Amplifier

The GAIN bit of the ADCNTRL2 register enables the Autozero Amplifier circuit. To perform a proper ADC conversion using the autozero amplifier, software must configure its non-inverting input (G4/AIN0) as a tri-state input bypassing the I/O circuitry. The autozero amplifier has a gain of 16, but is not defined as a true differential amplifier because the inverting SR_GND input must be connected as close to ground as possible (e.g. to act as a Kelvin connection) to reduce noise and improve the precision of the measurement.

To perform an ADC conversion through the autozero amplifier, the ACH1 input channel of the analog mux must be selected. The autozero ADC conversion is divided in three phases lasting a total of 20 conversion clock cycles. In the first phase, occupying the first six conversion cycles, it calculates the offset voltage of the amplifier. The second phase, occupying the next five cycles, adds or subtracts the offset voltage to the amplified input voltage which now has a gain of 16. The final phase, occupying the last nine cycles, converts the autozero input voltage to an 8-bit digital value and stores it in the ADATA register for easy access by software.

## 4.4 Uncommitted Amplifier

The Uncommitted Amplifier Enable (ENAMP) bit of the ADCNTRL2 register enables the Uncommitted Amplifier (AMP) circuit whose inverting input is connected to the G6/-$A_{IN}$ and output to the G7/$A_{OUT}$ port pins. Before enabling the AMP circuit, software must configure both the G6/-$A_{IN}$ and G7/$A_{OUT}$ pins as tri-state input ports bypassing all I/O circuitry. The AMP circuit may be used in any control or battery management applications.

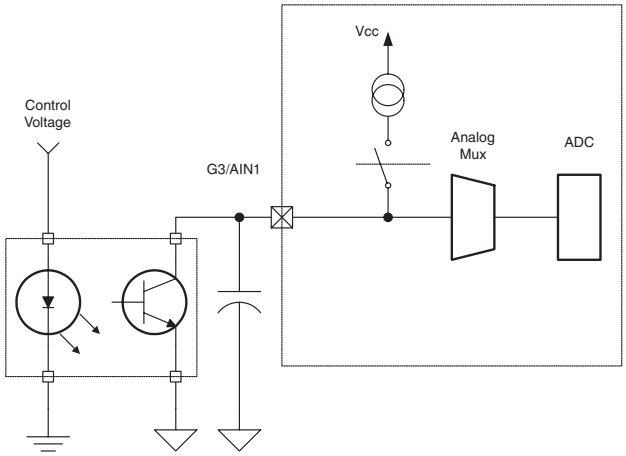
In control applications, the AMP circuit is used as the error amplifier in a hardware closed loop whose input connections are part of the external compensation loop circuit. The output of the amplifier ($A_{OUT}$) is internally fed to the Programmable Comparator circuit to control the PWM T1HS1 and T1HS2 inputs of the control plant block. An ADC conversion may be triggered to monitor $A_{OUT}$ by selecting the ACH5 input channel of the analog mux.

The AMP circuit may be configured as a general uncommitted amplifier whose non-inverting input is connected to $V_{REF}$. Therefore, the AMP circuit may only amplify differences with respect to $V_{REF}$. An ADC conversion may be triggered to convert the voltage at $A_{OUT}$ by selecting the ACH5 input channel of the analog mux. In battery management applications, the AMP circuit may be used to improve the resolution of the battery voltage measurement by adding a gain through the feedback loop. Voltage variation at a typical point will be amplified with a gain for better resolution, for example, to sense the Negative Delta V (NDV) to determine the end of change for a NiCD or NiMH battery.

## 4.5 Current Source Generator

The Current Source Enable (ENIS) bit of the ADCNTRL2 register enables the Current Source Generator (ISOURCE) circuit connected to the G3/AIN1 pin. Before enabling the ISOURCE circuit, software must configure the G3/AIN1 port as a tri-state input bypassing all I/O circuitry.[9] Once the ENIS bit is set, the ISOURCE circuit begins to generate $I_{SRC}$ of current typically used to interface to an opto-coupler output.[1] Figure 7 provides an example of a typical ISOURCE application where the voltage developed at the G3/AIN1 input can be converted by the ADC circuit if the ACH2 analog input channel is selected. The ISOURCE and ADC circuit combination may also be used to measure Capacitive Sensors or the resistance of a thermistor (NTC/PTC) to indirectly measure the temperature.

**Figure 7. Current Generator Interface**




1. Refer to the Electrical Characteristics section of the datasheet for details.
2. Refer to the PWM Timer 1 Circuit section of the datasheet for details regarding the ADSTROBE signal configuration.
3. Refer to Table 30 of the Device Memory section of the datasheet for the detailed memory map.
4. On the FMS7401L 8-pin device, the SR_GND is internally bonded to the GND pin.
5. Hardware interrupts are not executed by the microcontroller core unless the Global Interrupt enable (G) flag of the Status register is set. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
6. The ADC hardware interrupt will be executed in the defined priority order. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
7. Assuming the internal oscillator frequency is $F_{OSC}$=2MHz as specified in the Electrical Characteristics section of the datasheet.
8. The upper $F_{OSC}$ frequency (4MHz) is not a standard feature offered on the FMS7401L devices but is available upon request.
9. Refer to the I/O Ports section of the datasheet for details.

# 5 Programmable Comparator Circuit

The Programmable Comparator circuit is an analog comparator whose outputs may be monitored by software or fed into a digital delay filter used to disable the PWM Timer 1 circuit or its PWM cycle. The comparator's non-inverting input is software selectable by the COMPSEL bit of the ADCNTRL2 register.[1] If COMPSEL=0, the non-inverting input of the Programmable Comparator is the G4/AIN0 device pin. If COMPSEL=1, the non-inverting input of the Programmable Comparator is the G2/AIN2 device pin. Before enabling the Programmable Comparator circuit, the selected analog input port pin must be configured as a tri-state input bypassing the I/O circuitry.[2] The inverting input of the comparator is controlled by the Voltage Loop (VLOOP) enable bit of the comparator control (COMP) register. If VLOOP=0, the voltage loop is disabled and the inverting input of the analog comparator is configured as one of the 63 programmable voltage levels ($V_{THL}$, $V_{THU}$). If VLOOP=1, the analog comparator is set in a voltage loop configuration with the Uncommitted (Error) Amplifier output ($A_{OUT}$) connected to the comparator's inverting input (see Figure 9).

The Programmable Comparator circuit may be configured and controlled by software through the two 8-bit Comparator Control (COMP) and Digital Delay (DDELAY) registers. Both the Programmable Comparator and the digital delay filter must be enabled by software by setting the Comparator Enable (COMPEN) and clearing the EPWM bits of the Digital Delay (DDELAY) register. Upon a system reset, the Programmable Comparator is disabled and the digital delay filter is enabled. The COMP circuit is automatically disabled during Halt Mode. After exiting the Halt Mode, software must wait at least 10 instruction clock cycles before reading the COUT bit to ensure that the internal circuit has stabilized.

**Table 8. Programmable Comparator (COMP) Control Register Bit Definitions**

| COMP Register (addr. 0xA0) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| CL[5:0] | | | | | | VLOOP | COUT |

| Bit | Description |
|---|---|
| CL[5:0] | Programmable Comparator Voltage Reference Level bits. Refer to Table 9 and Table 10 for details. |
| VLOOP | (0) Configures the inverting input of the analog comparator as one of the 63 programmable voltage levels ($V_{THL}$, $V_{THU}$).<br>(1) Configures the analog comparator in a voltage loop configuration with the Uncommitted Amplifier output ($A_{OUT}$) connected to the inverting input. |
| COUT | (0) G2/AIN2 or G4/AIN0 non-inverting input is less than inverting input configured by VLOOP.<br>(1) G2/AIN2 or G4/AIN0 non-inverting input is greater than inverting input configured by VLOOP. |

## 5.1 Programmable Comparator's Voltage Threshold Levels (VLOOP=0)

The Programmable Comparator circuit is configured to compare the G4/AIN0 or G2/AIN2 non-inverting input against the programmable voltage threshold levels on its inverting input (see Table 9 and Table 10). The comparator output ($C_{OUT}$) is 1 when the G4/AIN0 or G2/AIN2 input pin rises above the selected voltage threshold. As long as the input stays above the selected voltage threshold, the $C_{OUT}$ signal will hold its state. The $C_{OUT}$ signal will equal zero if the G4/AIN0 or G2/AIN2 input voltage falls below the programmed threshold voltage or if the Programmable Comparator circuit is disabled. Software may change the programmed threshold voltage on-the-fly as needed in the application. If the digital delay filter circuit is enabled (EPWM=0), the $C_{OUT}$ signal is monitored for its rising edge to generate the PWMOFF signal. Refer to Figure 8 and the following Digital Delay Filter with PWMOFF Output section for addition details.

Bit 6 of the ADCNTRL2 register is the Programmable Comparator non-inverting input selection (COMPSEL) bit.[1] If COMPSEL=0, the non-inverting input of the Programmable Comparator is the G4/AIN0 device pin. If COMPSEL=1, the non-inverting input of the Programmable Comparator is the G2/AIN2 device pin. Before enabling the Programmable Comparator circuit, the selected analog input port pin must be configured as a tri-state input bypassing the I/O circuitry.[2]

Bits 7-2 (CL[5:0]) is the comparator voltage threshold level selection bits of the Comparator Control (COMP) register. The CL bits may be programmed to select one of the voltage threshold levels as the inverting input of the analog comparator. Refer to Table 9 and Table 10 for a detailed list of voltages.

Bit 1 of the Comparator Control (COMP) register is the Programmable Comparator circuit's voltage loop (VLOOP) configuration enable bit. If VLOOP=0, the Programmable Comparator circuit is configured to compare the analog G4/AIN0 or G2/AIN2 input (COMPSEL=0 or 1) to one of the 63 voltage threshold levels. If VLOOP=1, enables the voltage loop configuration where the analog G4/AIN0 or G2/AIN2 input (COMPSEL=0 or 1) to the Uncommitted (Error) Amplifier output ($A_{OUT}$).

Bit 7 of the Digital Delay (DDELAY) register is the Programmable Comparator circuit enable (COMPEN) bit. If COMPEN=0, the Programmable Comparator circuit is disabled and the $C_{OUT}$ signal is low. If COMPEN=1, the Programmable Comparator circuit is enabled and the $C_{OUT}$ signal generated by the comparison of the two inputs.

The comparator output ($C_{OUT}$) signal is latched by the main system instruction ($F_{ICLK}$) clock into bit 0 (COUT) of the Comparator Control (COMP) register. Software may only read the COUT bit to monitor the comparator's activity. The COUT bit cannot cause a microcontroller hardware interrupt or perform any other action.

**Figure 8. Programmable Comparator Block Diagram (VLOOP = 0)**

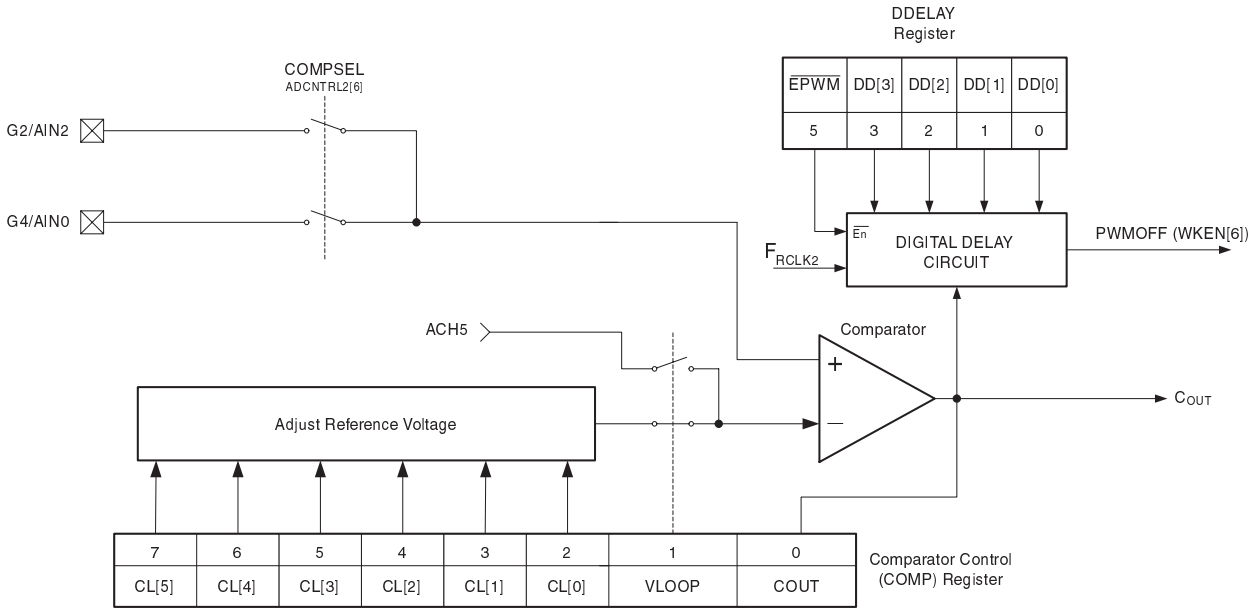


**Table 9. Programmable Comparator Lower Voltage Reference $V_{THL}$ (Levels 1 – 31)**

| Level | CL[5] | CL[4] | CL[3] | CL[2] | CL[1] | CL[0] | Voltage Reference |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 1 | 35mV |
| 2 | 0 | 0 | 0 | 0 | 1 | 0 | 50mV |
| 3 | 0 | 0 | 0 | 0 | 1 | 1 | 64mV |
| 4 | 0 | 0 | 0 | 1 | 0 | 0 | 78mV |
| 5 | 0 | 0 | 0 | 1 | 0 | 1 | 93mV |
| 6 | 0 | 0 | 0 | 1 | 1 | 0 | 107mV |
| 7 | 0 | 0 | 0 | 1 | 1 | 1 | 121mV |
| 8 | 0 | 0 | 1 | 0 | 0 | 0 | 135mV |
| 9 | 0 | 0 | 1 | 0 | 0 | 1 | 150mV |
| 10 | 0 | 0 | 1 | 0 | 1 | 0 | 164mV |
| 11 | 0 | 0 | 1 | 0 | 1 | 1 | 178mV |
| 12 | 0 | 0 | 1 | 1 | 0 | 0 | 192mV |
| 13 | 0 | 0 | 1 | 1 | 0 | 1 | 205mV |
| 14 | 0 | 0 | 1 | 1 | 1 | 0 | 219mV |
| 15 | 0 | 0 | 1 | 1 | 1 | 1 | 233mV |
| 16 | 0 | 1 | 0 | 0 | 0 | 0 | 247mV |
| 17 | 0 | 1 | 0 | 0 | 0 | 1 | 261mV |
| 18 | 0 | 1 | 0 | 0 | 1 | 0 | 274mV |
| 19 | 0 | 1 | 0 | 0 | 1 | 1 | 288mV |
| 20 | 0 | 1 | 0 | 1 | 0 | 0 | 302mV |
| 21 | 0 | 1 | 0 | 1 | 0 | 1 | 316mV |
| 22 | 0 | 1 | 0 | 1 | 1 | 0 | 330mV |
| 23 | 0 | 1 | 0 | 1 | 1 | 1 | 343mV |
| 24 | 0 | 1 | 1 | 0 | 0 | 0 | 358mV |
| 25 | 0 | 1 | 1 | 0 | 0 | 1 | 371mV |
| 26 | 0 | 1 | 1 | 0 | 1 | 0 | 385mV |
| 27 | 0 | 1 | 1 | 0 | 1 | 1 | 401mV |
| 28 | 0 | 1 | 1 | 1 | 0 | 0 | 413mV |
| 29 | 0 | 1 | 1 | 1 | 0 | 1 | 429mV |
| 30 | 0 | 1 | 1 | 1 | 1 | 0 | 443mV |
| 31 | 0 | 1 | 1 | 1 | 1 | 1 | 459mV |

**Table 10. Programmable Comparator Upper Voltage Reference $V_{THU}$ (Levels 32 – 63)**

| Level | CL[5] | CL[4] | CL[3] | CL[2] | CL[1] | CL[0] | Voltage Reference |
|---|---|---|---|---|---|---|---|
| 32 | 1 | 0 | 0 | 0 | 0 | 0 | 0.46V |
| 33 | 1 | 0 | 0 | 0 | 0 | 1 | 0.51V |
| 34 | 1 | 0 | 0 | 0 | 1 | 0 | 0.56V |
| 35 | 1 | 0 | 0 | 0 | 1 | 1 | 0.61V |
| 36 | 1 | 0 | 0 | 1 | 0 | 0 | 0.66V |
| 37 | 1 | 0 | 0 | 1 | 0 | 1 | 0.71V |
| 38 | 1 | 0 | 0 | 1 | 1 | 0 | 0.76V |
| 39 | 1 | 0 | 0 | 1 | 1 | 1 | 0.81V |
| 40 | 1 | 0 | 1 | 0 | 0 | 0 | 0.86V |
| 41 | 1 | 0 | 1 | 0 | 0 | 1 | 0.91V |
| 42 | 1 | 0 | 1 | 0 | 1 | 0 | 0.96V |
| 43 | 1 | 0 | 1 | 0 | 1 | 1 | 1.01V |
| 44 | 1 | 0 | 1 | 1 | 0 | 0 | 1.06V |
| 45 | 1 | 0 | 1 | 1 | 0 | 1 | 1.11V |
| 46 | 1 | 0 | 1 | 1 | 1 | 0 | 1.16V |
| 47 | 1 | 0 | 1 | 1 | 1 | 1 | 1.21V |
| 48 | 1 | 1 | 0 | 0 | 0 | 0 | 1.27V |
| 49 | 1 | 1 | 0 | 0 | 0 | 1 | 1.32V |
| 50 | 1 | 1 | 0 | 0 | 1 | 0 | 1.37V |
| 51 | 1 | 1 | 0 | 0 | 1 | 1 | 1.43V |
| 52 | 1 | 1 | 0 | 1 | 0 | 0 | 1.48V |
| 53 | 1 | 1 | 0 | 1 | 0 | 1 | 1.53V |
| 54 | 1 | 1 | 0 | 1 | 1 | 0 | 1.58V |
| 55 | 1 | 1 | 0 | 1 | 1 | 1 | 1.63V |
| 56 | 1 | 1 | 1 | 0 | 0 | 0 | 1.68V |
| 57 | 1 | 1 | 1 | 0 | 0 | 1 | 1.73V |
| 58 | 1 | 1 | 1 | 0 | 1 | 0 | 1.78V |
| 59 | 1 | 1 | 1 | 0 | 1 | 1 | 1.83V |
| 60 | 1 | 1 | 1 | 1 | 0 | 0 | 1.88V |
| 61 | 1 | 1 | 1 | 1 | 0 | 1 | 1.94V |
| 62 | 1 | 1 | 1 | 1 | 1 | 0 | 1.99V |
| 63 | 1 | 1 | 1 | 1 | 1 | 1 | 2.04V |

## 5.2 Hardware Voltage and Current Loop Control (VLOOP=1)

The Programmable Comparator circuit is configured to compare the G4/AIN0 or G2/AIN2 non-inverting input against the output of the Uncommitted (Error) Amplifier ($A_{OUT}$) when configured in a voltage/current loop control mode. In the voltage/current loop control, the inner (current) loop is performed by comparing the level on the G4/AIN0 or G2/AIN2 input against the voltage present at the Uncommitted (Error) Amplifier ($A_{OUT}$). The Uncommitted Amplifier performs the outer (voltage) loop control by detecting the error signal and driving the current control loop to modify the PWM duty cycle (see Figure 9). The FMS7401L voltage/current loop configuration can be used in SMPS applications where the digital loop control does not have the required accuracy and speed. Refer to the ADC Circuit section of the datasheet for the Uncommitted Amplifier configuration details.

When VLOOP=1, the comparator output ($C_{OUT}$) is 1 when the G4/AIN0 or G2/AIN2 input pin rises above $A_{OUT}$. As long as the input stays above $A_{OUT}$, the $C_{OUT}$ signal will hold its state. The $C_{OUT}$ signal will equal zero if the G4/AIN0 or G2/AIN2 input voltage falls below $A_{OUT}$ or if the Programmable Comparator circuit is disabled. If the digital delay filter circuit is enabled (EPWM=0), the $C_{OUT}$ signal is monitored for its rising edge to generate the PWMOFF signal. Refer to Figure 9 and the following Digital Delay Filter with PWMOFF Output section for addition details.

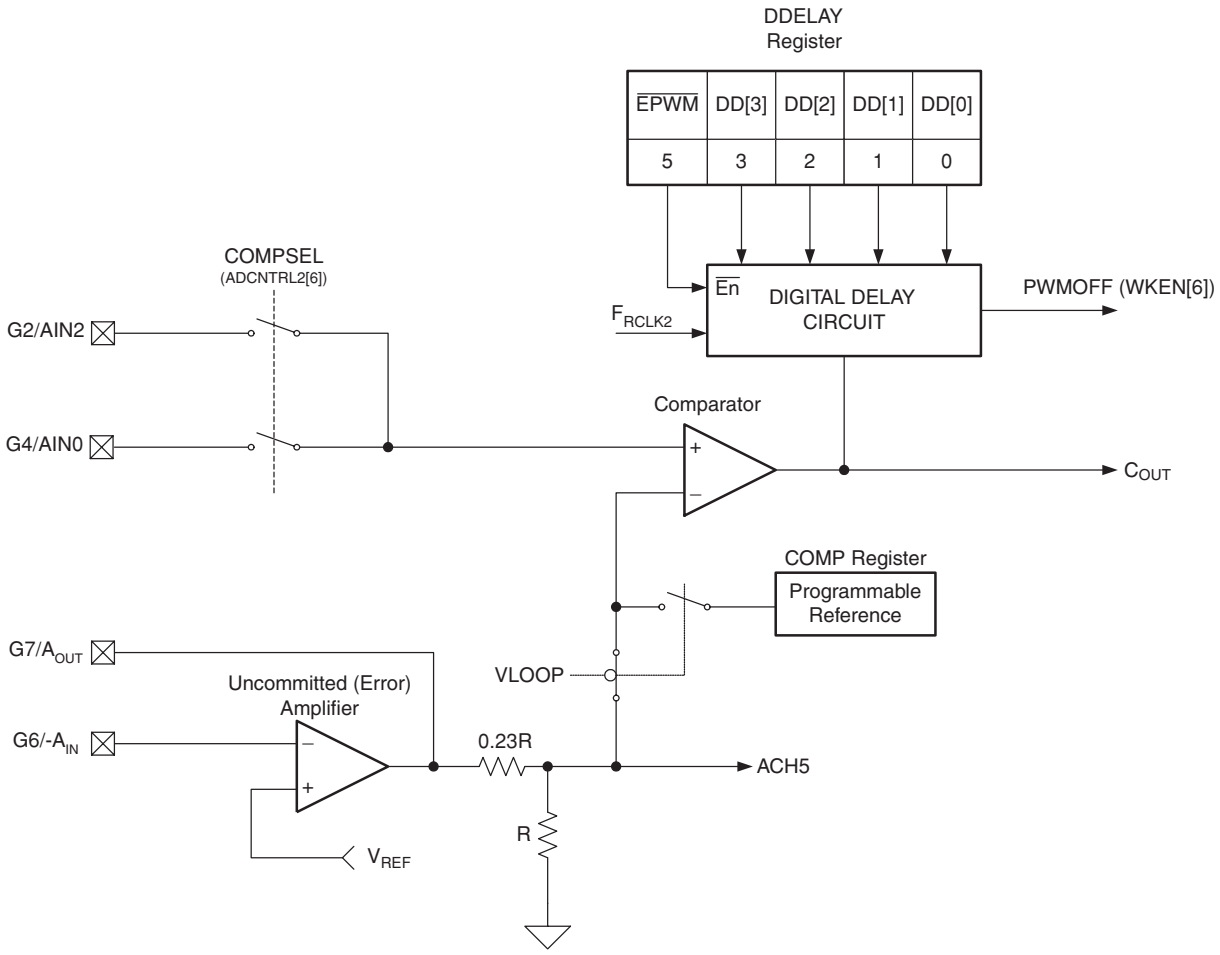
Bit 6 of the ADCNTRL2 register is the Programmable Comparator non-inverting input selection (COMPSEL) bit.[1] If COMPSEL=0, the non-inverting input of the Programmable Comparator is the G4/AIN0 device pin. If COMPSEL=1, the non-inverting input of the Programmable Comparator is the G2/AIN2 device pin. Before enabling the Programmable Comparator circuit, the selected analog input port pin must be configured as a tri-state input bypassing the I/O circuitry.[2]

Bit 1 of the Comparator Control (COMP) register is the Programmable Comparator circuit's voltage loop (VLOOP) configuration enable bit. If VLOOP=0, the Programmable Comparator circuit is configured to compare the analog G4/AIN0 or G2/AIN2 input (COMPSEL=0 or 1) to one of the 63 voltage threshold levels. If VLOOP=1, enables the voltage loop configuration where the analog G4/AIN0 or G2/AIN2 input (COMPSEL=0 or 1) to the Uncommitted (Error) Amplifier output ($A_{OUT}$).

Bit 7 of the Digital Delay (DDELAY) register is the Programmable Comparator circuit enable (COMPEN) bit. If COMPEN=0, the Programmable Comparator circuit is disabled and the $C_{OUT}$ signal is low. If COMPEN=1, the Programmable Comparator circuit is enabled and the $C_{OUT}$ signal generated by the comparison of the two inputs.

Bit 0 (COUT) of the Comparator Control (COMP) register is the latched comparator output ($C_{OUT}$) signal. If the Programmable Comparator circuit is enabled, the $C_{OUT}$ signal is latched by the main system instruction ($F_{ICLK}$) clock into the COUT bit of the COMP register. Software may only read the COUT bit to monitor the comparator's activity. The COUT bit cannot cause any microcontroller hardware interrupt or any other actions.

**Figure 9. Programmable Comparator Block Diagram (VLOOP = 1)**



## 5.3 Digital Delay Filter with PWMOFF Output

The Programmable Comparator's output ($C_{OUT}$) is fed into the digital delay filter with a programmable delay time. The $C_{OUT}$ signal toggles from 0 to 1 when the external input (G4/AIN0 or G2/AIN2) voltage is higher than the programmed voltage threshold or Uncommitted Amplifier output ($A_{OUT}$), depending on the state of VLOOP. The $C_{OUT}$ rising edge transition triggers the programmable digital delay counter to begin incrementing. With each digital delay count, its value is compared against the value stored in the DD[3:0] bits of the Digital Delay (DDELAY) control register. If $C_{OUT}$ remains high when the digital delay count equaling DD[3:0] completes, the PWMOFF signal transitions from 0 to 1. This rising edge transition of the PWMOFF signal is then used to either disable the PWM Timer 1 circuit completely or the current PWM cycle forcing the PWM output signals to their resting (off) state. The PWMOFF output signal may also be programmed as an input of the G6 port MIW circuit. Interrupts may be triggered if the G6 port MIW circuit is enabled and configured to trigger its microcontroller hardware interrupt (EDGEI). Refer to the Multi-input Wakeup Circuit section of the datasheet regarding for configuration details.

Bit 7 of the DDELAY register is the Programmable Comparator circuit enable (COMPEN) bit. If COMPEN=0, the Programmable Comparator circuit is disabled and the $C_{OUT}$ signal is low. If COMPEN=1, the Programmable Comparator circuit is enabled and the $C_{OUT}$ signal is generated by the comparison of the two inputs.

Bit 6 (PWMINT) of the DDELAY register, if set to 1, selects the PWMOFF signal in place of its G6 input to the MIW circuit. Software must then enable the MIW PWMOFF/G6 circuit by setting the WKEN[6] bit. The WKEDG[6] bit must also be cleared to select the rising edge transitions on the PWMOFF signal as its WKPND[6] bit trigger. Software may monitor the WKPND[6] flag or enable the MIW hardware interrupt (EDGEI) to help detect when the PWMOFF signal is triggered.[3]

Bit 5 (EPWM) of the DDELAY register is the digital delay filter and PWMOFF signal enable bit. The EPWM bit is active low so that on power-up (after a system reset) the digital delay filter circuit is automatically enabled once the Programmable Comparator circuit is enabled. If the Programmable Comparator and PWM Timer 1 circuits are enabled, since the filter is defaulted enabled, the PWMOFF signal may disable the PWM Timer 1 upon a comparator transition. If the digital delay filter and PWMOFF circuit is not needed, software must set the EPWM bit to 1 disabling the filter before enabling the Programmable Comparator to prevent unwanted disables of the PWM Timer 1 circuit or its outputs.

Bit 4 (OFFMODE) of the DDELAY register determines how the PWMOFF signal affects the PWM Timer 1 circuit. If OFFMODE=0 and the Timer 1 circuit is configured in an enabled PWM Mode, Timer 1 is automatically disabled forcing the PWM T1HS1 and T1HS2 output signals to their resting (off) states with the rising edge of the PWMOFF signal. The T1C0 bit of the T1CNTRL2 register is cleared, reinitializing the 12-bit TMR1 counter to 0x000. Software must re-enable the Timer 1 circuit to reactivate the PWM output signals. If OFFMODE=1 and Timer 1 is configured in PWM Mode, the T1HS1 and T1HS2 output signals are forced to their resting (off) state until the current PWM cycle completes. Once the PWM cycle completes, the PWMOFF signal releases the T1HS1 and T1HS2 output signals and they resume with their normal operation even if the $C_{OUT}$ signal remains active (1). The next PWMOFF trigger will not occur until the next rising edge of $C_{OUT}$.

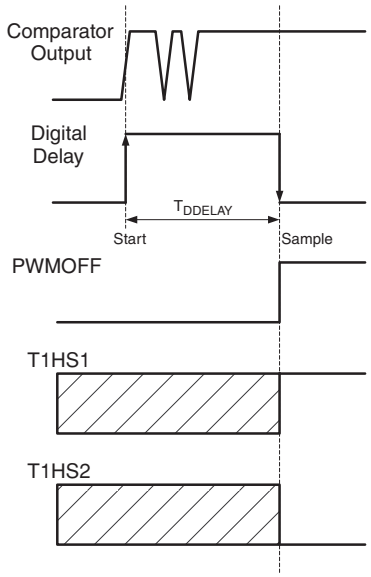
Bits 3-0 (DD[3:0]) of the DDELAY register determine how long to delay the trigger of the PWMOFF signal once the rising edge of $C_{OUT}$ has been detected. Once the digital delay counter is triggered, the delay count is compared against the value stored in the DD[3:0] bits. Once the delay counter completes its DD[3:0] count, if $C_{OUT}$ remains high, the PWMOFF signal is triggered. The digital delay counters increment at the device reference clock rate ($F_{RCLK2}$).[4]

### Table 11. Digital Delay (DDELAY) Register Bit Definitions

| DDELAY Register (addr. 0xA2) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| COMPEN | PWMINT | EPWM | OFFMODE | DD[3:0] | | | |

| Bit | Description |
|---|---|
| COMPEN | (0) Disable the Programmable Comparator circuit.<br>(1) Enable the Programmable Comparator circuit. |
| PWMINT | (0) The input of the G6 MIW circuit network is the G6/-$A_{IN}$ device pin.<br>(1) The input of the G6 MIW circuit network is the PWMOFF output signal (not the G6/-$A_{IN}$ device pin). |
| EPWM | (0) Enables the Digital Delay Filter circuit. The PWMOFF output is triggered by $C_{OUT}$ after the programmed delay ($T_{DDELAY}$).<br>(1) Disables the Digital Delay Filter circuit and the PWMOFF output signal. |
| OFFMODE | (0) PWM outputs switched off and Timer 1 stops after a comparator detection with delay.<br>(1) PWM outputs switched off for the current PWM cycle only. |
| DD[3:0] | Digital delay after $C_{OUT}$ triggers high, $T_{DDELAY}$ = DD • (1/$F_{RCLK2}$) |

**Figure 10. Digital Delay Timing**




1. Refer to the ADC Circuit section of the datasheet for additional details.
2. Refer to the I/O Ports section of the datasheet for details.
3. Hardware interrupts are not executed by the microcontroller core unless the Global Interrupt enable (G) flag of the Status register is set. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
4. Refer to the Clock Circuit section of the datasheet for details regarding the $F_{RCLK2}$ clock.

# 6 PWM Timer 1 Circuit

The Pulse Width Modulation (PWM) Timer 1 circuit, a programmable 12-bit PWM timer with a 3-bit prescaler can be configured to operate in both PWM and Input Capture Modes. In PWM Mode, the Timer 1 circuit may be configured to generate pulses of a specified duty cycle and period on the T1HS1 (G0), T1HS2 (G5), and/or ADSTROBE (G1) timer output ports. On the other hand, in Input Capture Mode, the Timer 1 circuit may be configured to capture and store the current timer value at the time of a trigger defined by the rising or falling edge of the T1HS2 (G5) input port. In addition, the T1HS1 and ADSTROBE PWM outputs may be generated as in the PWM Mode.

The Timer 1 circuit backbone is a 12-bit programmable up-counter (TMR1) that is accessible by software, with read-only access, through the 4-bit TMR1HI and 8-bit TMR1LO memory mapped registers where TMR1={TMR1HI, TMR1LO}.[1] Upon a system reset or once entering a new Timer 1 mode of operation, the 12-bit TMR1 counter is initialized to 0x000. Once a selected Timer 1 mode is enabled, the TMR1 counter will begin incrementing by the $F_{T1CLK}$ clock with the programmed divide factor defined by a 3-bit prescaler. Once the TMR1 overflows, TMR1 is reinitialized to 0x000 and resumes incrementing until software disables the mode. In PWM Mode, the TMR1 overflow value may be programmed by software where, in Input Capture Mode, the TMR1 will overflow once the 0xFFF count completes. The Timer 1 circuit may be programmed to generate microcontroller hardware interrupts with every TMR1 overflow and capture.

The Timer 1 $F_{T1CLK}$ clock may be programmed to operate from high to low frequencies with the use of the programmable digital clock multiplier (PLL) and internal oscillator in order to provide the maximum flexibility for various PWM applications.

## 6.1 PWM Timer 1 Configuration Registers

Software must access the six memory mapped PWM Timer 1 registers to configure and control the Timer 1 circuit.[1] The 8-bit Prescale (PSCALE) register is used to configure the entire FMS7401L clock structure including the Timer 1's $F_{T1CLK}$. The 12-bit Timer 1 Compare A (T1CMPA), Timer 1 Compare B (T1CMPB), and Timer 1 Reload (T1RA) registers are used to define the PWM output signal's duty cycle and period. The 5-bit Dead Time (DTIME) register is used to define the time delay ($T_{DT}$) between the PWM T1HS1 and T1HS2 edge transitions while in PWM Mode. The Timer 1 Control (T1CNTRL) register is used to select Timer 1's operating mode, enable its PWM output signals, and control its hardware interrupt (TMRI1).

### 6.1.1 PSCALE Register and Timer 1 Clock Configuration

Although the PSCALE register is part of the PWM Timer 1 circuit, its register bits configure the clock structure for the entire FMS7401L along with the Timer 1 clock ($F_{T1CLK}$). Refer to the Clock Circuit section of the datasheet for details regarding the device's clock structure. The $F_{T1CLK}$ source may be supplied by either the programmable $F_{PWMCLK}$ PLL output or the main instruction ($F_{ICLK}$) clock. Once the $F_{T1CLK}$ source is selected, it may not be changed while the Timer 1 circuit is in PWM mode and running or in Input Capture Mode (run mode). Upon a system reset, the PSCALE register is automatically initialized to 0x00.

Bit 7 of the PSCALE register is the PLL circuit enable (PLLEN) bit. Before using any of the PLL outputs, software must enable the PLL circuit and wait the $T_{PLL_LOCK}$ to ensure that the PLL is locked into its appropriate frequency and in phase. The PLLEN bit may not be changed while the Timer 1 circuit is in run mode. Any attempts to write to PLLEN under this condition will be ignored and its value will remain unchanged.

Bits 6 and 5 (FS[1:0]) of the PSCALE register are the bits used to select between the different output frequencies available to the PLL's $F_{PWMCLK}$ output signal. FS selects between the four available PLL divide factors (divide-by-1/2/4/8) selecting an output frequency of 8/16/32/64MHz (see Table 13). The FS bits may be changed by software at any time; however, if the Timer 1 circuit is in run mode, the FS value will not change the $F_{PWMCLK}$ output frequency until after the TMR1 counter overflows ending the current PWM cycle. The last FS value at the TMR1 counter overflow will dictate the divide factor of the $F_{PWMCLK}$ output for the next PWM cycle. When reading FS, the value reported will be the last value written by software and may not necessarily reflect the divide factor for the current PWM cycle.

Bit 4 of the PSCALE register is the frequency selection (FSEL) bit for the Timer 1 circuit. FSEL is used to select between the slow or high frequency options, ultimately selecting the $F_{T1CLK}$ to be sourced either by the $F_{ICLK}$ or $F_{PWMCLK}$ (see Table 13). If FSEL=0, the slow frequency option is selected and the $F_{ICLK}$ will then source the $F_{T1CLK}$ with either a 1/8MHz frequency determined by the FMODE bit, as discussed later in the section. If FSEL=1, the high frequency option is selected and the $F_{PWMCLK}$ will then source the $F_{T1CLK}$ at a frequency selected by the FS[1:0] bits. The FSEL bit may not be set if the PLL is not enabled (PLLEN=0) or changed while the Timer 1 circuit is in run mode. Any attempts to write to FSEL under this condition will be ignored and its value will remain unchanged.

Bit 3 (FMODE) of the PSCALE register is the frequency selection bit for the main instruction clock ($F_{ICLK}$). FMODE is used to select between the slow or high frequency options, ultimately selecting the $F_{ICLK}$ to be sourced either by the internal oscillator (or the external digital clock) operating at $F_{OSC}$[2] or the PLL's $F_{(FS=0)}$ output signal.[3] If FMODE=0, the slow frequency option is selected and the internal oscillator will then source the $F_{ICLK}$ at a $F_{OSC}/2$ frequency. If FMODE=1, the high frequency option is selected and the $F_{(FS=0)}$ will then source the $F_{ICLK}$ with PLL's divide-by-8 output frequency. With the FMODE bit enabled, it is possible to execute instructions at a speed approximately eight times faster than the standard. The FMODE bit may not be set if the PLL is not enabled (PLLEN=0). Any attempts to write to FMODE while PLLEN=0 will force FMODE=0 ignoring any set instructions. Once the PLL has been enabled, software may change $F_{ICLK}$'s clock source on-the-fly during normal instruction execution in order to speed-up a particular action.

Bits 2-0 of the PSCALE register are the three prescaler (PS[2:0]) bits used to divide the $F_{T1CLK}$ to obtain a wider frequency range on the PWM output signals (see Table 14). The PS bits are used by the Timer 1 circuit to increment the 12-bit TMR1 at a frequency equal to $F_{T1CLK}$ divided-by 1 through 8. The PS bits (like FS) may be changed by software at any time; however, if the Timer 1 circuit is in run mode, the PS value will not change the prescale division factor until after the TMR1 counter overflows ending the current PWM cycle. The last PS value at the TMR1 counter overflow will dictate the prescale divide factor of the $F_{T1CLK}$ for the next PWM cycle. When reading PS, the value reported will be the last value written by software and may not necessarily reflect the divide factor for the current PWM cycle.

**Table 12. Prescale (PSCALE) Register Bit Definitions**

| PSCALE Register (addr. 0xA4) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| PLLEN | FS[1:0] | | FSEL | FMODE | PS[2:0] | | |

| Bit | Description |
|---|---|
| PLLEN | (0) Disables the PLL circuit.<br>(1) Enables the PLL circuit. |
| FS[1:0] | PLL Divide Factor Selection Bits. Refer to Table 13 for details. |
| FSEL | (0) Selects $F_{ICLK}$ as Timer 1's clock ($F_{T1CLK}$) source.<br>(1) Selects $F_{PWMCLK}$ PLL output as Timer 1's clock ($F_{T1CLK}$) source. |
| FMODE | (0) Selects $F_{CLK}$ divided-by-2 output as the main system instruction clock ($F_{ICLK}$) source.<br>(1) Selects $F_{(FS=0)}$ PLL output as the main system instruction clock ($F_{ICLK}$) source. |
| PS[2:0] | Timer 1 Prescale Selection Bits. Refer to Table 13 for details. |

**Table 13. PLL Divide Factor Selection Bits and the $F_{T1CLK}$ Resolution ($F_{OSC}$=2 MHz)**

| FS[1:0] | | $F_{PWMCLK}$ | $F_{T1CLK}$ (FSEL=0) | | $F_{T1CLK}$ (FSEL=1) | Max PWM Freq. (8-bit resolution) | | Max PWM Freq. (12-bit resolution) | |
|---|---|---|---|---|---|---|---|---|---|
| | | | FMODE=0 | FMODE=1 | | FSEL=0 FMODE=0 | FSEL=1 | FSEL=0 FMODE=0 | FSEL=1 |
| 0 | 0 | 8 MHz | 1 MHz | 8 MHz | 8 MHz | 3.906 kHz | 31.25 kHz | 244.14 Hz | 1.95 kHz |
| 0 | 1 | 16 MHz | 1 MHz | 8 MHz | 16 MHz | 3.906 kHz | 62.5 kHz | 244.14 Hz | 3.9 kHz |
| 1 | 0 | 32 MHz | 1 MHz | 8 MHz | 32 MHz | 3.906 kHz | 125 kHz | 244.14 Hz | 7.8 kHz |
| 1 | 1 | 64 MHz | 1 MHz | 8 MHz | 64 MHz | 3.906 kHz | 250 kHz | 244.14 Hz | 15.625 kHz |

**Table 14. Timer 1 Prescale Selection (PS) Bits**

| PS[2] | PS[1] | PS[0] | Timer 1 Clock |
|---|---|---|---|
| 0 | 0 | 0 | $F_{T1CLK}$ / 1 |
| 0 | 0 | 1 | $F_{T1CLK}$ / 2 |
| 0 | 1 | 0 | $F_{T1CLK}$ / 3 |
| 0 | 1 | 1 | $F_{T1CLK}$ / 4 |
| 1 | 0 | 0 | $F_{T1CLK}$ / 5 |
| 1 | 0 | 1 | $F_{T1CLK}$ / 6 |
| 1 | 1 | 0 | $F_{T1CLK}$ / 7 |
| 1 | 1 | 1 | $F_{T1CLK}$ / 8 |

### 6.1.2 PWM Cycle Configuration Registers

The PWM Timer 1 circuit has three 12-bit (T1CMPA, T1CMPB, T1RA) and one 5-bit (DTIME) configuration registers used to specify the duty cycle and period of the Timer 1 output signals. Upon a system reset, all four registers are initialized with ones (0xFFF, 0x1F). All configuration registers must be programmed with their appropriate values prior to enabling the Timer 1 circuit. Except for the DTIME register, the T1CMPA, T1CMPB and T1RA registers may be changed by software at any time; however, if the Timer 1 circuit is in run mode, the register values will not change the Timer 1 output signal's attributes until after the TMR1 counter overflows ending the current PWM cycle. The last register values at the TMR1 counter overflow will dictate the output signal's attributes for the next PWM cycle. When reading the registers, the value reported will be the last value written by software and may not necessarily reflect the output signal's attributes for the current PWM cycle.

The 12-bit T1RA is Timer 1's reload/capture register (depending on the selected operating mode). All 12 bits are accessible by software through the 4-bit T1RAHI and 8-bit T1RALO memory mapped registers where T1RA= {T1RAHI, T1RALO}.[1] In PWM Mode, T1RA configures the period of the T1HS1, T1HS2 and ADSTROBE outputs and determines after what TMR1 count it will overflow (reinitialize to 0x000) to begin the next PWM cycle. In Input Capture Mode, T1RA contains the captured value of the TMR1 count at the time of the trigger defined by the rising or falling edge of the T1HS2 input.

The 12-bit T1CMPA is Timer 1's Compare A register that dictates the length of the resting (off) state of the T1HS1 and T1HS2 output signals. All 12 bits are accessible by software through the 4-bit T1CMPAHI and 8-bit T1CMPALO memory mapped registers where T1CMPA={T1CMPAHI, T1CMPALO}.[1] In PWM Mode, the TMR1 counter is compared against T1CMPA (TMR1=T1CMPA) to determine when the first transition of the T1HS1 and T1HS2 output signals should be triggered. In Input Capture Mode, the T1CMPA value is still used to configure T1HS1 but has no affect on the T1HS2 signal since it is used as an input of the Timer 1 circuit in this mode. Software must ensure that the total T1CMPA plus $T_{DT}$ time is not greater than or equal to the total T1RA plus $T_{DT}$ times.

The 12-bit T1CMPB is Timer 1's Compare B register that dictates the length of the resting (off) state of the ADSTROBE output signal. All 12 bits are accessible by software through the 4-bit T1CMPBHI and 8-bit T1CMPBLO memory mapped registers where T1CMPB={T1CMPBHI, T1CMPBLO}.[1] The TMR1 counter is compared against T1CMPB to determine when the first transition of the ADSTROBE output signal should be triggered. If enabled, at the rising edge of ADSTROBE (TMR1=T1CMPB) an Analog-to-Digital Converter (ADC) conversion may be initiated.[4] Software must ensure that the total T1CMPB plus $T_{DT}$ time is not greater than or equal to the total T1RA plus $T_{DT}$ times.

Bits 4-0 (DT[4:0]) of the DTIME register determines the amount of dead time ($T_{DT}$) delay, if any, between the T1HS1 and T1HS2 output signal transitions (see Table 13). In PWM Mode, once the TMR1 counter equals the T1CMPA value, the T1HS1 signal transitions ending its resting (off) state. The dead time delay counter is then initiated incrementing with each edge of the $F_{T1CLK}$ up to the programmed DT[4:0] value.[5] Once $T_{DT}$ has passed, the T1HS2 signal will then transition ending its resting (off) state. As the TMR1 counter continues, once the TMR1 counter equals the T1RA value, the T1HS2 signal transitions ending its active (on) state. The dead time delay counter is then reinitiated incrementing to the DT[4:0] value. Once $T_{DT}$ has passed, the T1HS1 signal will then transition ending its active (on) state.

**Table 15. Dead Time (DTIME) Register Bit Definitions**

| DTIME Register (addr. 0xA5) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| X | X | X | DT[4:0] | | | | |

| DT[4:0] | Dead Time Delay ($T_{DT}$) |
|---|---|
| 0x00 | No Dead Time Delay |
| 0x01 – 0x1F | $(1/F_{T1CLK}) \cdot DT$ |

### 6.1.3 Timer 1 Control Register

The Timer 1 Control (T1CNTRL) register is used to select Timer 1's operating mode, enable its PWM output signals, and control its hardware interrupt (TMRI1). Upon a system reset, the T1CNTRL register is automatically initialized to 0x00. Refer to Table 16 and Table 17 for additional details regarding the T1CNTRL register bits.

Bit 7 (T1C3) of the T1CNTRL register selects between the two Timer 1 operating modes. If T1C3=0, the Timer 1 circuit will be configured in PWM mode. Once Timer 1 is configured in PWM Mode, the TMR1 counter must be started in order for the circuit to be enabled and considered to be in run mode. If T1C3=1, the Timer 1 circuit will be configured in Input Capture Mode. Once in Input Capture Mode, the Timer 1 circuit is immediately enabled and the TMR1 counter will automatically begin incrementing from its initial state (0x000) until Timer 1's operating mode is returned to a disabled PWM Mode (its default). Therefore, software must issue a write command clearing T1C3 and T1C0 within the same instruction. If T1C3 is cleared while T1C0 is set, the Timer 1 circuit will remain in Input Capture Mode until the TMR1 overflows and $T_{DT}$ completes ending the current PWM cycle. Once the TMR1 overflows, the operating mode is switched to a disable PWM Mode even though T1C0=1. Software must clear T1C0 before re-enabling the Timer 1 circuit in any of the two operating modes.

Bit 6 (T1C2) of the T1CNTRL register has two functions depending on Timer 1's selected operating mode. In PWM Mode, T1C2 is used to enable the T1HS2 (G5) output signal (if set). Otherwise, the T1C2 bit (if zero) disables the T1HS2 output signal. The T1HS2 signal on the G5 pin may be configured as an active high or low output depending on the configured PORTGD configuration. If PORTGD[5]=0, the T1HS2 (G5) output is active high, otherwise it is active low. In Input Capture Mode, T1C2 is used to select the edge to trigger a TMR1 T1HS2 input capture. If T1C2=0, every rising edge of the T1HS2 input will trigger a TMR1 capture. If T1C2=1, every falling edge will trigger a capture. Software may change the value of T1C2 at any time; however, if the circuit is in run mode, T1C2 will not change the circuit's attribute until after the TMR1 counter overflows and the $T_{DT}$ passes completing the current PWM cycle. The last T1C2 value after the $T_{DT}$ completes, will dictate the circuit's attribute for the next PWM cycle. When reading the T1C2, the value reported will be the last value written by software and may not necessarily reflect the circuit's attribute for the current PWM cycle.

Bit 5 (T1C1) of the T1CNTRL register enables or disable the T1HS1 (G0) output signal for either operating mode. If T1C1=1, the T1HS1 output signal is enabled, otherwise it is disabled. The T1HS1 signal on the G0 pin may be configured as an active high or low output depending on the configured PORTGD configuration. If PORTGD[0]=0, the T1HS1 (G0) output is active high, otherwise it is active low. Software may change the value of T1C1 at any time; however, if the circuit is in run mode, T1C1 will not change the circuit's attribute until after the TMR1 counter overflows and the $T_{DT}$ passes completing the current PWM cycle. The last T1C1 value after the $T_{DT}$ completes, will dictate the circuit's attribute for the next PWM cycle. When reading the T1C1, the value reported will be the last value written by software and may not necessarily reflect the circuit's attribute for the current PWM cycle.

The ADSTROBE signal is outputted through the device's G1 pin when bit 0 (T1BOUT) of the T1CNTRL register is set to 1. The ADSTROBE output is always generated by the Timer 1 circuit regardless of the state of T1BOUT. The ADSTROBE signal on the G1 pin may be configured as an active high or low output depending on the configured PORTGD configuration. If PORTGD[1]=0, the ADSTROBE (G1) output is active high, otherwise it is active low. Software may change the value of T1BOUT at any time; however, if the circuit is in run mode, T1BOUT will not change the device's I/O attribute until after the TMR1 counter overflows and the $T_{DT}$ passes completing the current PWM cycle. The last T1BOUT value after the $T_{DT}$ com-

pletes, will dictate the device's I/O attribute for the next PWM cycle. When reading the T1BOUT, the value reported will be the last value written by software and may not necessarily reflect the device's I/O attribute for the current PWM cycle.

Bit 4 (T1C0) of the T1CNTRL register has two functions depending on Timer 1's selected operating mode. In PWM Mode, when T1C0=1, the TMR1 circuit becomes enabled and begins to increment from its initial 0x000 state; otherwise, the TMR1 counter is stopped and reinitialized. Software may disable the Timer 1 circuit at any time; however, the TMR1 counter and PWM outputs will not be disabled until the current PWM cycle completes. Software should monitor the T1C0 bit to determine when the PWM cycle ends and Timer 1 circuit actually disabled. In Input Capture Mode, the T1C0 bit is one of the TMR1 overflow (a transition from 0xFFF to 0x000) pending flags used to trigger the Timer 1 Circuit's hardware interrupt if the interrupt is enabled. In order for software to properly monitor the TMR1 overflows, the T1C0 bit must be cleared before the next TMR1 overflow.

Bit 3 (T1PND) of the T1CNTRL register has two functions depending on Timer 1's selected operating mode. In either operating modes, the T1PND bit is one of the Timer 1 Circuit's hardware interrupt pending flags if the interrupt is enabled. In PWM Mode, the T1PND bit is triggered by a TMR1 overflow (a transition from the T1RA count to 0x000). However, in Input Capture Mode, the T1PND bit is triggered by the capture of the current TMR1 value by the rising or falling edge of the T1HS2 (G5) input port. In order for software to properly monitor the pending flag, the T1PND bit must be cleared before the next TMR1 overflow or capture.

Bit 2 of the T1CNTRL register is the Timer 1's microcontroller hardware interrupt enable (T1EN) bit. If set, hardware interrupts are enabled and trigger by the T1PND and/or T1C0 pending flags depending on Timer 1's operating mode.[6] If in PWM Mode, the hardware interrupt is triggered only by the T1PND bit. If in Input Capture Mode, the T1PND and T1C0 bits are logically-ORed together. As long as a Timer 1 pending flag is set, the hardware interrupt will continue to execute software's Timer 1 interrupt service routine until the pending flag is cleared.[7]

The SBIT or RBIT instructions may be used to either set or clear one of the T1CNTRL register bits, like the T1EN bit. The SBIT and RBIT instructions both take two instruction clock cycles to complete their execution. In the first cycle, all register bits are automatically read to obtain their most current value. In the second cycle, the bit to be set/cleared is given its new value and all bits are then re-written to the register. Using the SBIT/RBIT instruction to set/clear an enable bit with a pending flag in the same register may cause a potential hazard. Software may inadvertently clear a recently triggered pending flag if the trigger happened during the second phase of the SBIT/RBIT instruction execution. To avoid this condition, the LD instruction must be used to set or clear the interrupt enable bits. The Timer 1 circuit is designed such that software may not trigger a pending flag by writing a 1 to the T1PND and T1C0 (if in Input Capture Mode) bits, they may only be cleared. The action of writing a 1 to a T1PND and T1C0 register bits holds the current bit values. The action of writing a 0 to the T1PND and T1C0 register bits clears the bit values. Therefore, if Timer 1 is configured for a rising edge triggered input capture mode with outputs enabled and software is to enable interrupts without interrupting the pending flags, the "LD T1CNTRL, #0BDH" instruction should be used. The T1EN bit will be set to 1 without clearing T1PND and/or T1C0.

**Table 16. Timer 1 Control (T1CNTRL) Register Bit Definitions**

| T1CNTRL Register (addr. 0xAE) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| T1C3 | T1C2 | T1C1 | T1C0 | T1PND | T1EN | X | T1BOUT |

| Bit | Description |
|---|---|
| T1C3 | Timer 1 Mode Configuration Bit. Refer to Table 17 for details. |
| T1C2 | Timer 1 Mode Configuration Bit. Refer to Table 17 for details. |
| T1C1 | Timer 1 Mode Configuration Bit. Refer to Table 17 for details. |
| T1C0 | PWM Mode<br>(0) Stop the PWM Timer 1 circuit.<br>(1) Start the PWM Timer 1 circuit. |
| | Input Capture Mode<br>(0) Timer 1's TMR1 overflow pending flag is cleared.<br>(1) Timer 1's TMR1 overflow pending flag is triggered. |
| T1PND | PWM Mode<br>(0) Timer 1's TMR1 overflow pending flag is cleared.<br>(1) Timer 1's TMR1 overflow pending flag is triggered. |
| | Input Capture Mode<br>(0) Timer 1 capture pending flag is cleared.<br>(1) Timer 1 capture pending flag is triggered. |
| T1EN | (0) Disables Timer 1 hardware interrupts.<br>(1) Enables Timer 1 hardware interrupts. |
| T1BOUT | (0) Retain normal I/O function of the G1/AIN3 pin.<br>(1) Enables Timer 1's ADSTROBE output to be sent to the G1 output port. |

**Table 17. Timer 1 Mode Configuration Bits**

| T1C3 | T1C2 | T1C1 | Timer Mode Source | Interrupt | Timer count on |
|---|---|---|---|---|---|
| 0 | 0 | 0 | PWM mode no output toggle | TMR1 Overflow | Prescaler Input |
| 0 | 1 | 1 | PWM mode T1HS1 and T1HS2 toggle | TMR1 Overflow | Prescaler Input |
| 0 | 0 | 1 | PWM mode T1HS1 toggle | TMR1 Overflow | Prescaler Input |
| 0 | 1 | 0 | PWM mode T1HS2 toggle | TMR1 Overflow | Prescaler Input |
| 1 | 0 | 0 | Capture mode no T1HS1 toggle | TMR1 Overflow<br>T1HS2 rising-edge | Prescaler Input |
| 1 | 0 | 1 | Capture mode with T1HS1 toggle | TMR1 Overflow<br>T1HS2 rising-edge | Prescaler Input |
| 1 | 1 | 0 | Capture mode no T1HS1 toggle | TMR1 Overflow<br>T1HS2 falling-edge | Prescaler Input |
| 1 | 1 | 1 | Capture mode with T1HS1 toggle | TMR1 Overflow<br>T1HS2 falling-edge | Prescaler Input |

## 6.2 Pulse Width Modulation (PWM) Mode

In PWM Mode, the Timer 1 circuit may be configured to generate pulses of a specified duty cycle and period on the T1HS1 (G0), T1HS2 (G5), and/or ADSTROBE (G1) timer outputs. The 12-bit TMR1 counter increments at the $F_{T1CLK}$ clock rate defined by the FSEL bit of the PSCALE register. Refer to the previous PSCALE Register and Timer 1 Clock Configuration section of the datasheet for details.

A PWM cycle begins with the TMR1 counter incrementing from 0x000 until it matches the value stored in the T1RA register. At this point, the TMR1 counter completes its T1RA count and overflows (a transitions from T1RA to 0x000) setting the T1PND flag of the T1CNTRL register ending the PWM cycle. The Timer 1 circuit has two additional TMR1 compare (T1CMPA and T1CMPB) registers used to generate the T1HS1, T1HS2, and ADSTROBE output signals. All three output signals are initialized to a resting (off) state. Once the TMR1 counter is enabled (by setting the T1C0 bit of the T1CNTRL register), both compare registers are matched against the incrementing TMR1 counter. When the TMR1 completes its count equal to the value stored in the T1CMPA and T1CMPB registers, the T1HS1, T1HS2, and ADSTROBE output signals are set to an active (on) state until the TMR1 counter matches the value stored in the T1RA compare register (overflows). Once the TMR1 counter overflows, the output signals are cleared returning them to a resting (off) state. Refer to Figure 11 for a Timer 1 PWM Mode block diagram.

The PWM Timer 1 can be programmed to toggle one or both PWM output signals (T1HS1 and T1HS2) to support a variety of output configurations (half bridge, full bridge,[8] low side or high side driving). These outputs may be used to drive an external half-bridge driver and are enabled by programming the T1C1 and T1C2 bits in the T1CNTRL register (see Table 16). The T1HS1 (G0) and T1HS2 (G5) output signals may be configured with opposite phases and dead time controlled edges (see Figure 12). The phases of the output signals are configured by the bits of the PORTGD I/O configuration register.[9] Upon device power-up, the T1HS1 and T1HS2 signals may be programmed to default as active high/low outputs by the default I/O configuration register bits in the non-volatile Initialization Register 4.[10] Both G0/T1HS1 and G5/T1HS2 pins will configure to their programmed default state after $T_{DIO}$[2] from the system reset trigger (e.g. from a POR). The G0/T1HS1 and G5/T1HS2 pins may both be configured as outputs with common or opposite phases. If configured as outputs, the PORTGD[0] and PORTGD[5] bits configure the T1HS1 and T1HS2 signals as active high or low. If the PORTGD bit is 0, the output signal is active high, otherwise it is active low. The PORTGD[1] bit also configures the G1/ADSTROBE pin as an active high/low signal once configured as an output. The Initialization Register 4 bits only default the G0/T1HS1 and G5/T1HS2 pins not the G1/ADSTROBE pin. From factory, the G0/T1HS1, G5/T1HS2 and G1/ADSTROBE device pins are defaulted as tri-stated inputs. The pins must be configured by the Initialization Register 4 bits or by software directly through the PORTGC and PORTGD register as an output port before enabling the TMR1 counter and its outputs.

The dead time counter of the Timer 1 circuit controls the dead time ($T_{DT}$) delay between the T1HS1 and T1HS2 output edge transitions through the DT[4:0] bits of the DTIME register. The dead time counter delay is first triggered after the TMR1 counter equals to the T1CMPA value and the T1HS1 signal transitions from its resting (off) to its active (on) state. Once the programmed $T_{DT}$ completes, the T1HS2 signal then transitions from its resting (off) to its active (on) state. The dead time counter delay is triggered for a second time after the TMR1 counter equals to the T1RA value and the T1HS2 signal transitions from its active (on) to its resting (off) state. Once the programmed $T_{DT}$ completes, the T1HS1 signal then transitions from its active (on) to its resting (off) state ending the PWM cycle. The PWM cycle is considered complete once the TMR1 counter completes the T1RA count plus $T_{DT}$ even in the T1HS1 and T1HS2 outputs are disabled.

The T1HS1 and T1HS2 PWM output signals may be programmed to be automatically disabled by the output of the digital filter (PWMOFF) in Programmable Comparator circuit. The output may be programmed to disable the Timer 1 circuit completely or disable only the current PWM cycle. Refer to the Programmable Comparator Circuit section of the datasheet for details.

The Timer 1's ADSTROBE output signal may be configured as the G1/ADSTROBE device output if the T1BOUT bit of the T1CNTRL register is set. The ADSTROBE signal, however, is always generated by the Timer 1 circuit. Initially, the ADSTROBE begins its PWM cycle at its resting (off) state and transitions to its active (on) state once the TMR1 counter completes its count equal to the T1CMPB value. The active (on) edge transition of the ADSTROBE output may be programmed to automatically trigger an ADC conversion cycle if the ENDAS bit of the ADCNTRL2 register is set. Refer to the ADC Circuit section of the datasheet for details.

The T1PND bit of the T1CNTRL register is set once the TMR1 counter completes the count equal to the T1RA value (overflows). Software may use the T1PND bit to monitor the PWM cycles and/or trigger microcontroller hardware interrupts (TMRI1) if the T1EN bit of the T1CNTRL register is set. Software must clear the T1PND bit in order to detect a new overflow condition and/or trigger a new interrupt.[6]

**Figure 11. Timer 1's PWM Mode Block Diagram**

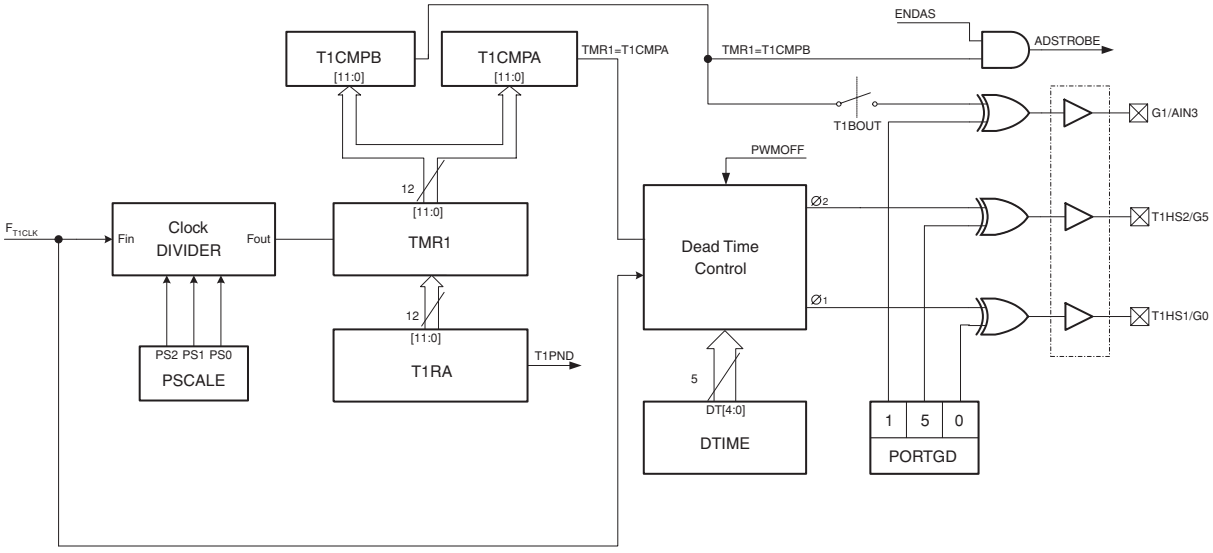
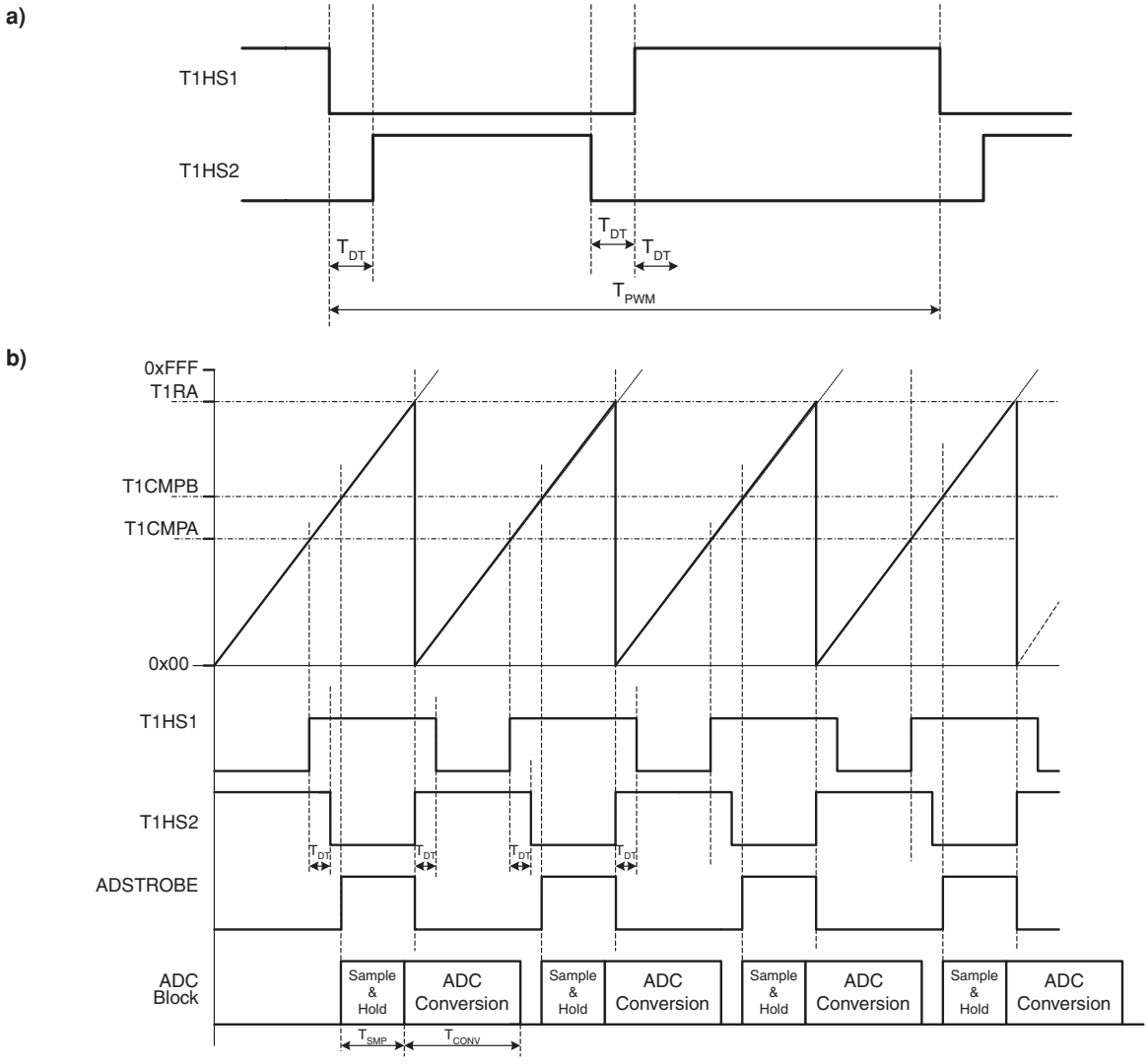



**Figure 12. Example PWM Output Signals a) and b)**



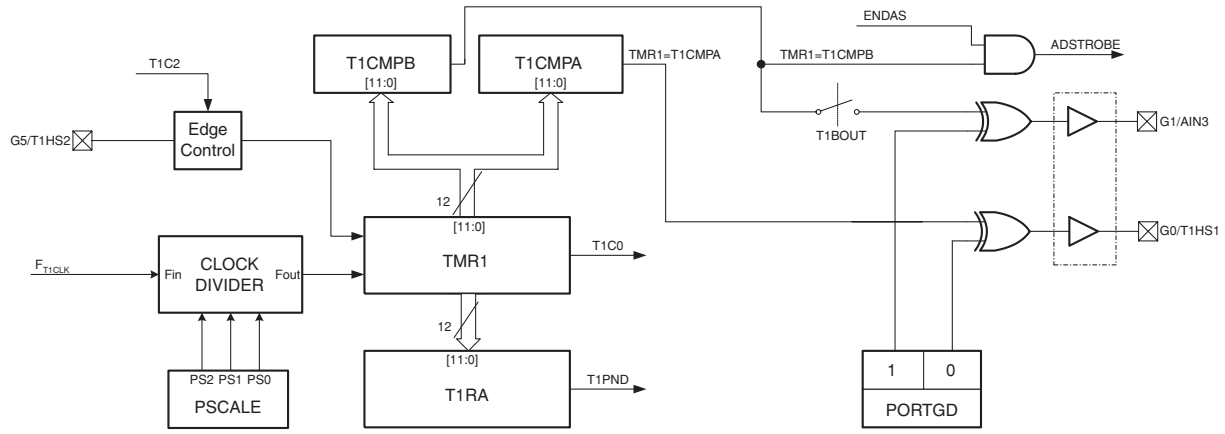
## 6.3 Input Capture Mode

When the PWM Timer 1 circuit is configured in Input Capture Mode, the T1HS2 signal is used as an input of the Timer 1 circuit instead of an output as in PWM Mode. The G5/T1HS2 device pin should be configured by software as an input port. The Timer 1 circuit may be programmed to capture the TMR1 counter value in the T1RA register with every rising or falling edge transition of the T1HS2 input. With each TMR1 capture, the T1PND bit of the T1CNTRL register is set. For synchronization purposes, the T1HS2 input is synchronized with the $F_{T1CLK}$ clock before being allowed to trigger a TMR1 capture. A maximum of three $F_{T1CLK}$ cycle TMR1 capture delay will occur with each edge transition on the G5/T1HS2 device pin.

Once the Timer 1 circuit is configured for Input Capture Mode, the TMR1 counter starts incrementing continuously from 0x000 through 0xFFF until the Timer 1 is returned to a disabled PWM operating mode. Once the TMR1 counter overflows (transitions from 0xFFF to 0x000) the T1C0 pending bit on through T1CNTRL register is set.

In Input Capture Mode, it is still possible to generate output signals with variable duty-cycle thanks to the T1CMPA and T1CMPB compare registers. If enabled, the T1HS1 and ADSTROBE output signals may be generated as in PWM Mode. Refer to the previous Pulse Width Modulation (PWM) Mode section of the datasheet for details.

**Figure 13. Timer 1's Input Capture Mode Block Diagram**




1. Refer to Table 30 of the Device Memory section of the datasheet for the detailed memory map.
2. Refer to the Electrical Characteristics section of the datasheet.
3. The PLL's ($F_{(FS=0)}$) output is not affected by the FS[1:0] bit value of the PSCALE register and merely shares the FS[1:0]=00 divide factor.
4. Refer to the ADC Circuit section of the datasheet for additional details.
5. The three PS bits have no affect on the dead time, only the TMR1 counter.
6. Hardware interrupts are not executed by the microcontroller core unless the Global Interrupt enable (G) flag of the Status register is set. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
7. The Timer 1 hardware interrupt will be executed in the defined priority order. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
8. The full bridge requires two additional output ports to complete the bridge configuration.
9. Refer to the I/O Ports section of the datasheet for additional details.
10. Refer to the Device Memory section of the datasheet for details regarding the initialization registers.

# 7 Timer 0 Circuit

Timer 0's main circuit is a 12-bit free running up-counter whose clock source is the main system instruction clock ($F_{ICLK}$). The main counter may be used to generate microcontroller hardware interrupts and serve as a prescaler for the Idle and Watchdog Timers.

After power-up or any system reset, the Timer 0's 12-bit counter is initialized to 0x000 and continuously increments with each instruction clock. The 12-bit counter is not memory mapped; therefore, software cannot read or write to the counter registers. However, software may monitor the Timer 0's main counter by reading the state of the Timer 0 Pending (T0PND) bit of the Timer 0 Control (T0CNTRL) memory mapped register.[1] The T0PND flag is automatically set with each counter overflow (a transition from 0xFFF to 0x000) which occurs after every 4,096 cycles. At every overflow, the counter rolls over to 0x000 and continues to increment. In order for software to properly monitor the main counter, the T0PND bit must be cleared before the next counter overflow.

The T0CNTRL register houses two hardware interrupt enable bits. The WKINTEN register bit is the MIW hardware interrupt enable bit. For details regarding its usage refer to the Multi-input Wakeup Circuit section of the datasheet. The T0INTEN register bit is Timer 0's microcontroller hardware interrupt (TMRI0) enable bit. If set, hardware interrupts are enabled and trigger by the T0PND flag.[2] As long as a Timer 0 pending flag is set, the hardware interrupt will continue to execute software's Timer 0 interrupt service routine until the pending flag is cleared.[3]

The SBIT or RBIT instructions may be used to either set or clear the T0INTEN or WKINTEN bits. The SBIT and RBIT instructions both take two instruction clock cycles to complete their execution. In the first cycle, all register bits are automatically read to obtain their most current value. In the second cycle, the bit to be set/cleared is given its new value and all bits are then re-written to the register. Using the SBIT/RBIT instruction to set/clear an enable bit with a pending flag in the same register may cause a potential hazard. Software may inadvertently clear a recently triggered pending flag if the trigger happened during the second phase of the SBIT/RBIT instruction execution. To avoid this condition, the LD instruction must be used to set or clear the interrupt enable bits. The Timer 0 circuit is designed such that software may not trigger a pending flag by writing a 1 to a T0PND register bit, it may only be cleared. The action of writing a 1 to a T0PND register bit holds the current bit value. The action of writing a 0 to a T0PND register bit clears the bit value. Therefore, the "LD T0CNTRL, #083H" instruction will set both interrupt enable bits without clearing T0PND.

**Table 18. Timer 0 Control (T0CNTRL) Register Definitions[4]**

| T0CNTRL Register (addr. 0xB6) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| WKINTEN | x | x | x | x | x | T0PND | T0INTEN |

## 7.1 Idle Timer

Once the device enters Idle Mode, the microcontroller core and other main circuits are disabled for current conservation. The Idle Timer will automatically wake the device from Idle Mode, if the MIW has not already done so, after a maximum of 8,192 cycles.[5]

The Idle Timer is a 1-bit extension of the Timer 0's main 12-bit up-counter. With each overflow of the main counter, the Idle Timer extension bit is toggled essentially causing the Idle Timer overflow to occur after 8,192 cycles. Once the Idle Timer overflow flag is triggered, the device wakes from Idle Mode and starts its instruction execution with the next clock cycle.

The Idle Timer overflow flag cannot be monitored by software. Therefore, in order to maximize the time that the device remains in Idle Mode software must monitor the T0PND flag. Once the T0PND flag is triggered, software may then issue the Idle Mode command. Software may also loop on the Idle Mode command to extend the average time the device remains in Idle Mode, thereby reducing the overall current consumption.

## 7.2 Watchdog Timer

The Watchdog Timer is used to safely recover the device in the rare event of a processor "runaway condition" by issuing a system reset. A Watchdog Timer runs continuously with Timer 0's main 12-bit up-counter; however, a Watchdog Reset will not

issue a system reset unless the feature is enabled by the Watchdog Enable (WDEN) bit of the Initialization Register 1.[6] The WDEN bit can only be set while the device is in programming mode.[7] If set, the Watchdog Timer's system reset ability will always power-up enabled. Software cannot disable Watchdog Resets. The Watchdog Reset can only be disabled in programming mode by clearing the WDEN bit as long as the memory write protect (WDIS) feature is not enabled.

The Watchdog Timer is a 4-bit extension of the Timer 0's main 12-bit up-counter. With each overflow of the main counter, the Watchdog Timer extension bits may increment to a count of 16. If the Watchdog Timer is allowed to increment to the 16$^{th}$ count, a Watchdog Reset is issued triggering a system reset. The system reset will initialize all device circuits and instruction execution will restart at the default program counter address (0xC00) after $T_{RESET}$ delay.[8]

In order to service the Watchdog Timer to prevent a reset, software must write 0x1B to the Watchdog Service Register (WDSVR) before every 61,440 cycles (the 16$^{th}$ Watchdog Timer count) and not earlier than the first 4,096 cycles (the 1$^{st}$ Watchdog Timer count) since the last Watchdog Timer service or system reset. Once the Watchdog Timer is serviced, the 4-bit Watchdog Timer is cleared and then continues to increment. The Watchdog Timer will issue a reset if it is serviced too frequently or not frequently enough where the servicing of the Watchdog Timer is controlled completely by software.

The Watchdog Timer, like Timer 0's counter, is not memory mapped and cannot be accessed by software. Software must monitor the Watchdog Timer by keeping a count of the number of T0PND flags since the last service or reset. If software clears the T0PND flags before the next Timer 0 counter overflow, software may count the number of triggered T0PND flags in order to determine when to next service the Watchdog Timer. The Watchdog Timer remains operational during Idle Mode; therefore, software should service the Watchdog Timer prior to entering Idle Mode to prevent false resets.

It is not recommended to service the Watchdog Timer within an interrupt service routine (ISR). For example, the most obvious place to issue the Watchdog service command seems to be within the Timer 0 ISR since it guarantees the Watchdog Timer service to occur within the allowed 4,096-61,440 cycle window. However, this action takes place automatically since the Timer 0 circuit runs independently from the microcontroller program execution without knowledge of the state of the microcontroller core. If the program execution is stuck in an infinite loop due to some unforeseen circumstances, the TMRI0 hardware interrupt will still be triggered executing software's ISR, the Watchdog Timer will then be serviced, and program execution will return to the infinite loop once the ISR completes. The infinite loop or "runaway condition" will continue undetected defeating the purpose of the Watchdog Reset feature. The ISRs may be used to keep track of the number of cycles since the last Watchdog service (e.g. keep a count of the number of T0PND flags triggered). However, the actual Watchdog service command must be issued within software's main program code.

**Table 19. Watchdog Service Register (WDSVR) Definition**

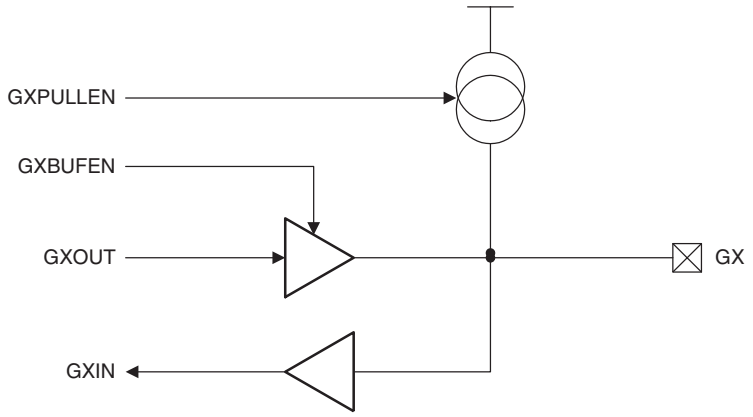
| WDSVR (addr. 0xB5) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |

1. Refer to Table 30 of the Device Memory section of the datasheet for the detailed memory map.
2. Hardware interrupts are not executed by the microcontroller core unless the Global Interrupt enable (G) flag of the Status register is set. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
3. The Timer 0 hardware interrupt will be executed in the defined priority order. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
4. After a system reset, the T0CNTRL register is defaulted to 0x00.
5. Refer to the Power Saving Modes section of the datasheet for Idle Mode wakeup conditions.
6. Refer to the Device Memory section of the datasheet for details regarding the Initialization Registers.
7. The FMS7401L must be placed in a special programming mode in order to have full write and read access of all of the device memories. Refer to the In-circuit Programming Specification section of the datasheet for details.
8. Refer to the Electrical Characteristics section of the datasheet for details.

# 8 I/O Ports

The eight I/O pins (six on the 8-pin package option) are bi-directional (see Figure 14). The bi-directional I/O pins can be individually configured by software to operate as high-impedance inputs, as inputs with weak pull-up, or as push-pull outputs. The operating state is determined by the contents of the corresponding bits in the data and configuration registers. Each bi-directional I/O pin can be used for general purpose I/O, or in some cases, for a specific alternate function determined by the on-chip hardware.

**Figure 14. PORTGD Logic Diagram**




## 8.1 I/O Registers

The I/O pins (G0–G7) have three memory mapped port registers associated with the I/O circuitry: a Port Configuration (PORTGC), Port Data (PORTGD) and Port Input (PORTGP) register.[1] PORTGC is used to configure the pins as inputs or outputs. A pin may be configured as an input by writing a 0 or as an output by writing a 1 to its corresponding PORTGC bit. If a pin is configured as an output, its PORTGD bit represents the state of the pin (1 = logic high, 0 = logic low). If the pin is configured as an input, its PORTGD bit selects whether the pin is a weak pull-up or a high-impedance input. Table 20 provides details of the port configuration options. The port configuration and data registers can both be read from or written to. Reading PORTGP returns the value of the port pins regardless of how the pins are configured. Since this device supports MIW, all input ports have Schmitt triggers.

Upon power-up, the PORTGC and PORTGD registers are initialized to 0x00. However, the G0/T1HS1 and G5/T1HS2 pins may be defaulted to the different I/O configurations defined by the default I/O configuration bits of the Initialization Register 4. Refer to Table 29 in the Device Memory section of the datasheet for details.

**Table 20. I/O Register Bit Assignments**

| PORTGC, PORTGD, PORTGD Registers (addr. 0xB3, 0xB2, 0xB4) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| G7[2] | G6[2] | G5 | G4 | G3 | G2 | G1 | G0 |

**Table 21. I/O Configuration Options**

| PORTGC Bit | PORTGD Bit | Port Pin Configuration |
|---|---|---|
| 0 | 0 | High-impedance input (tri-state input) |
| 0 | 1 | Input with pull-up (weak one input) |
| 1 | 0 | Push-pull zero output |
| 1 | 1 | Push-pull one output |

1. Refer to Table 30 of the Device Memory section of the datasheet for the detailed memory map.
2. Available only on the 14-pin package option.

# 9 Multi-input Wakeup Circuit

The Multi-input Wakeup (MIW) circuit may be used to wake the device from either Halt or Idle Mode[1] with an external event, generate flags for software monitoring and microcontroller hardware interrupts by any one or all I/O ports (G0–G7). The MIW circuit is configured using the Wakeup Enable (WKEN), Wakeup Edge (WKEDG), Wakeup Pending (WKPND) and T0CNTRL memory mapped registers.[2] The WKEN, WKEDG and WKPND are 8-bit registers where each bit corresponds to an I/O port pin (see Table 21). All four registers are initialized to 0x00 upon a system reset.

The PWMOFF output signal may also be programmed as an input of the G6 port MIW circuit. Interrupts may be triggered if the PWMOFF/G6 input MIW circuit is enabled and configured to trigger its microcontroller hardware interrupt (EDGEI). Bit 6 (PWMINT) of the DDELAY register, if set to 1, selects the PWMOFF signal in place of its G6 input to the MIW circuit. Software must then enable the MIW PWMOFF/G6 circuit by setting the WKEN[6] bit. The WKEDG[6] bit must also be cleared to select the rising edge transitions on the PWMOFF signal as its WKPND[6] bit trigger. Software may monitor the WKPND[6] flag or enable the MIW hardware interrupt (EDGEI) to help detect when the PWMOFF signal is triggered. Refer to the Programmable Comparator Circuit sections of the datasheet for addition details.

## 9.1 MIW Configuration Registers

The Wakeup Enable (WKEN) register individually enables an I/O port's edge transition to trigger a wakeup/interrupt pending flag. If the WKEN register bit is 1, the corresponding I/O port's MIW circuitry (defined by its bit number) is enabled; otherwise, the port circuitry remains disabled and the pending flag may not be triggered.

The Wakeup Edge (WKEDG) register bits are used to program an enabled I/O port's pending flag to be triggered from either a rising-/falling-edge transition. If the WKEDG register bit is 1, a falling-edge transition of the enabled I/O port will trigger the pending flag. If zero, a rising-edge transition of the enabled I/O port will trigger the pending flag.

The MIW circuit shares a single hardware interrupt (EDGEI) among all pending flags and is enabled by the Wakeup Interrupt enable (WKINTEN) bit of the T0CNTRL register.[2] The WKINTEN bit enables hardware interrupts for the MIW circuit if set to 1.[3]

The Wakeup Pending (WKPND) register contains the pending flags corresponding to each of the I/O port pins. If a WKPND register bit is 1, the programmed I/O port edge transition has triggered its pending flag. If zero, the flag is not pending and no transition has occurred from the last pending reset. A pending flag may only be triggered by enabled I/O ports (if its WKEN register bit is 1). Once a pending flag is triggered, all flags are logically-ORed together to trigger a WAKEOUT if in Halt/Idle Mode and/or hardware interrupts (if enabled). If software is to re-enter Halt/Idle Mode, all pending flags must be cleared, otherwise the command is ignored. Since all MIW pending flags share a single hardware interrupt, software must take care with the handling of the pending flags when more than one pending flag is enabled. As long as a MIW pending flag is set, the hardware interrupt will continue to execute software's MIW interrupt service routine with highest priority until all pending flags are cleared.[4]

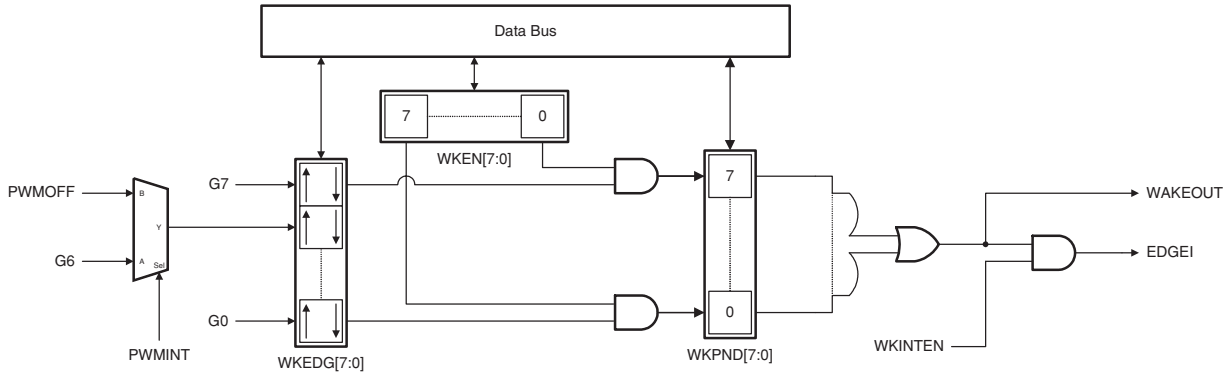
Upon exiting Halt/Idle Mode or before leaving software's MIW interrupt service routine, the RBIT instruction may be used to clear a particular pending flag. The RBIT instruction takes two instruction clock cycles to complete its execution. In the first cycle, all eight register bits are automatically read to obtain their most current value. In the second cycle, the bit to be cleared is given its new value and all bits are then re-written to the register. Using the RBIT instruction to clear an individual pending flag causes no potential hazards if only one wakeup I/O port is enabled. However, if more than one I/O port is enabled software may inadvertently clear a recently triggered pending flag if the trigger happened during the second phase of the RBIT instruction execution. To avoid this condition, the LD instruction must be used to clear a set pending flag. The MIW circuit is designed such that software may not trigger a pending flag by writing a 1 to a WKPND register bit, it may only be cleared. The action of writing a 1 to a WKPND register bit holds the current bit value. The action of writing a 0 to a WKPND register bit clears the bit value. Therefore, the "LD WKPND, #0F7H" instruction will clear the WKPND[3] while all others bits remain the same.

The MIW circuit can be used with the I/O ports configured as both an input and output. The MIW configuration and function is the same for both I/O configurations. However, when using the MIW circuit to wake the device from Halt/Idle Mode the wakeup I/O port must be configured as an input, otherwise the device will never exit the mode.

**Table 22. Multi-input Wakeup (MIW) Register Bit Assignments**

| WKEN, WKEDG, WKPND Registers (addr. 0xB1, 0xAF, 0xB0) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| G7[5] | PWMOFF[6]/G6[5] | G5 | G4 | G3 | G2 | G1 | G0 |

**Figure 15. Multi-input Wakeup (MIW) Block Diagram[6]**




1. Refer to the Power Saving Modes section of the datasheet for detail regarding Halt and Idle Mode.
2. Refer to Table 30 of the Device Memory section of the datasheet for the detailed memory map.
3. Hardware interrupts are not executed by the microcontroller core unless the Global Interrupt enable (G) flag of the Status register is set. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
4. No other hardware interrupts will be executed, aside from the software interrupt instruction, until the MIW hardware interrupt is no longer executed. Refer to the 8-Bit Microcontroller Core section of the datasheet for details.
5. Available only on the 14-pin package option.
6. The PWMOFF and PWMINT signals are the outputs from the Programmable Comparator's Digital Filter circuit. Refer to Programmable Comparator Circuit section of the datasheet for details.

# 10 8-Bit Microcontroller Core

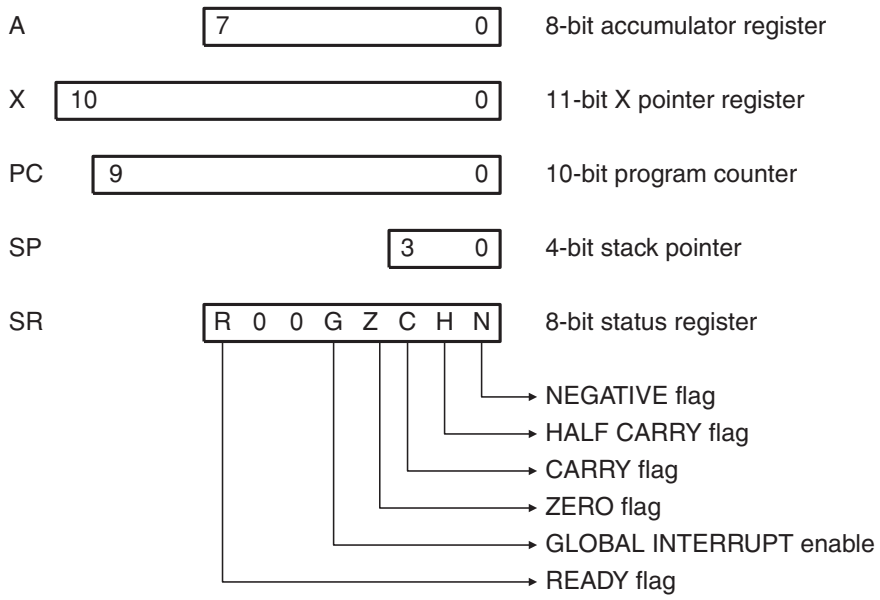
The FMS7401L's 8-bit microcontroller core is specifically designed for low cost applications involving bit manipulation, shifting and block encryption. It is based on a modified Harvard architecture meaning peripheral, I/O and RAM locations are addressed separately from instruction data.

The core differs from the traditional Harvard architecture by aligning the data and instruction memory sequentially. This allows the X-pointer (11-bits) to point to any memory location in either segment of the memory map. This modification improves the overall code efficiency of the microcontroller core and takes advantage of the flexibility found on the von Neumann architecture and stored program concept.

## 10.1 Core Registers

The microcontroller core has five general-purpose registers. These registers are the Accumulator (A), X-Pointer (X), Program Counter (PC), Stack Pointer (SP) and Status Register (SR). The X, SP and SR registers are memory mapped while A and PC are not.

**Figure 16. Core Program Model**



### 10.1.1 Accumulator (A)

The Accumulator is a general-purpose 8-bit register that is used to hold data and results of arithmetic calculations or data manipulations.

### 10.1.2 X-Pointer (X)

The X-Pointer register allows for an 11-bit indexing value to be added to an 8-bit offset creating an effective address used for reading and writing among the memory space. This provides software with the flexibility of storing lookup tables in the code EEPROM memory space for the core's accessibility during normal operation.[1]

The microcontroller core allows software to access the entire 11-bit X-Pointer register using the special X-pointer instructions e.g. LD X, #040H (see Table 24). Software may also access the register through any of the memory mapped instructions using the XHI (X[10:8]) and XLO (X[7:0]) variables located at address 0xBE and 0xBF (see Table 30).

The X register is divided into two sections. The most significant bit (MSB) is write only and selects between the data (0x000 to 0x0FF) or program (0xC00 to 0xFFF) memory space. The 10 least significant bits (LSBs) represent the specific address location within the data or program memory space.

For example: If X[10] = 0, the LD A, [#0,X] instruction will take the data at address X[9:0] from the data memory space (0x000 to 0x0FF) and load it into A. However, if X[10] = 1 the LD A, [#0,X] instruction will take the data at address X[9:0] from the program memory space (0xC00 to 0xFFF) and load it into A.

The X register can also serve as a counter or temporary storage register. However, this is true only for the 10-LSBs since the MSB is dedicated for memory space selection.

### 10.1.3 Program Counter (PC)

The 10-bit Program Counter (PC) register contains the address of the next instruction to be executed. After a system reset, PC is initialized to 0xC00 and the microcontroller core begins executing the instruction program residing in the code EEPROM memory at the initialized PC value.

### 10.1.4 Stack Pointer (SP)

The microcontroller core has an automatic program stack with a 4-bit stack pointer. The stack can be initialized to any location between addresses 0x30-0x3F in SRAM. Normally, the stack pointer is initialized by one of the first instructions in an application program. After a reset, the stack pointer is defaulted to 0xF pointing to the top of the stack at address 0x3F.

The stack is configured as a data structure which decrements from high to low memory. Each time a new address is pushed onto the stack, the microcontroller core decrements the stack pointer by two. Each time an address is pulled from the stack, the microcontroller core increments the stack pointer is by two. At any given time, the stack pointer points to the next free location in the stack.

When a subroutine is called by a jump-to-subroutine (JSR) instruction, the instruction's address is automatically pushed onto the stack with the least significant byte first. When the subroutine is finished, a return-from-subroutine (RET) instruction is executed. The RET instruction pulls the previously stacked return address and loads it into the program counter. Instruction execution then continues at the pulled return address.

### 10.1.5 Status Register (SR)

The 8-bit Status Register (SR) contains four condition code indicators (C, H, Z, and N), a global interrupt (G) mask bit, and the data EEPROM write ready (R) flag. The condition codes are automatically updated by most instructions (see Table 25). All status register bits except for the global interrupt mask are read only when using direct, indirect, or indexed instructions. The carry and half carry bits may be written by using their special inherent (SC, RC, LDC, RRC and RLC) instructions. Software cannot restore SR using the traditional microcontroller methods. Refer to the Interrupt Handling section for additional details.

### Carry/Borrow (C)

The carry flag is set if the arithmetic logic unit (ALU) performs a carry or borrows during an arithmetic operation and by its special inherent instructions—set carry (SC), load carry (LDC) and invert carry (INVC). The rotate instructions—rotate right/left through carry (RRC/RLC)—operate with and through the carry bit to facilitate multiple-word shift operations. The RC, SC, INVC, LDC and STC (store carry) instructions facilitate direct bit manipulation using the carry flag.

### Half Carry (H)

The half carry flag indicates whether an overflow has taken place on the boundary between the two nibbles in the accumulator. It is primarily used for Binary Coded Decimal (BCD) arithmetic calculation. The RC and SC instructions facilitate direct bit manipulation of the half carry flag.

### Zero (Z)

The zero flag is set if the result of an arithmetic, logic, or data manipulation operation is zero. Otherwise, the zero flag is cleared.

### Negative (N)

The result from an arithmetic, logic, or data manipulation operation is negative if the MSB is one, therefore setting the negative flag. Otherwise, the negative flag is cleared.

### Global Interrupt Mask (G)

The global interrupt bit (G) is a global mask that disables all maskable interrupt sources. If G is cleared, interrupts can become pending but the operation of the core continues uninterrupted (even if the individual interrupts are enabled). However, if G is set when an interrupt becomes pending the core will be interrupted and execute the appropriate interrupt service routine.

After a reset, G is defaulted to zero and can only be set by a software instruction. When an interrupt is recognized, G is automatically cleared after the program counter value is stacked and the interrupt vector addressing the interrupt service routine is fetched. Once the interrupt is serviced, a return-from-interrupt (RETI) instruction is normally executed to restore the program counter to the value before the interrupt occurred. G is the restored to one after the return from interrupt is executed. Although G can be set within an interrupt service routine, “nesting” interrupts in this way should only be done when there is a clear understanding of latency and of the arbitration mechanism.

**Table 23. Interrupt Priority Sequence**

| Priority (5 highest, 1 lowest) | Interrupt | |
|---|---|---|
| 5 | Software | (INTR) |
| 4 | MIW | (EDGEI) |
| 3 | Timer 0 | (TMRI0) |
| 2 | PWM Timer 1 | (TMRI1) |
| 1 | ADC | (ADCI) |

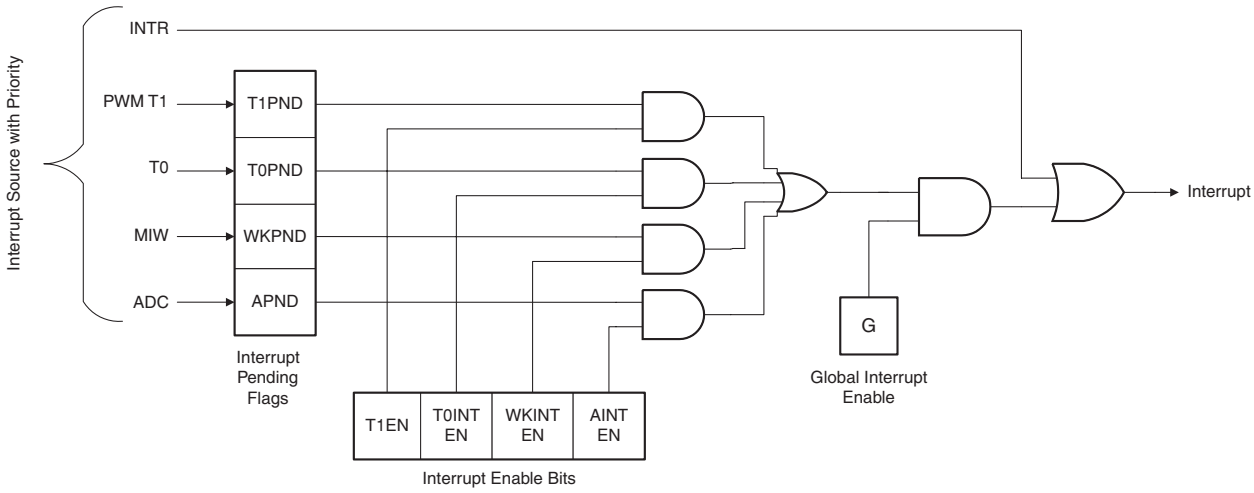
### 10.1.6 Interrupt Handling

When an interrupt is recognized, the current instruction completes its execution. The return address (the current value in the program counter, PC) is pushed onto the stack, the global interrupt (G) mask of the status register (SR) is cleared, and execution continues at the address specified by the respective interrupt vector (see Table 30). This process takes five instruction cycles to complete. The interrupt vector contains the address of the software's interrupt service routine (ISR). Initially, the ISR may save (if necessary) the status register's contents. Software, however, cannot restore SR using the traditional microcontroller methods. Just before ending the ISR, software may restore the contents of SR by using only the special inherent instructions (e.g. SC, RC and LDC) or specially defined software routines since all SR bits except for G are read only when using direct, indirect, or indexed instructions (e.g. LD , ST, RBIT or SBIT). Upon exiting the ISR, software must clear the appropriate triggering pending flag and execute a return-from-interrupt (RETI) instruction. The RETI instruction pulls the saved return address off the stack in reverse order restoring PC and setting G of SR to one. Instruction execution resumes at the restored the program counter address.

The microcontroller core is capable of supporting five interrupts. Four are maskable through G of the SR and the fifth (software interrupt) is not inhibited by G (see Figure 17). The execution of the INTR instruction generates a software interrupt. Once the INTR instruction is executed, the microcontroller core will interrupt whether G is set or not. The INTR interrupt is executed in the same manner as the other maskable interrupts where PC is stacked and G is cleared. This means, if G was enabled prior to the software interrupt the RETI instruction must be used to return from interrupt in order to restore G to one. However, if G was not enabled prior to the software interrupt the RET instruction must be used.

In the case of simultaneous multiple interrupts, the microcontroller core prioritizes the interrupts. See Table 23 for the interrupt service priority sequence.

**Figure 17. Basic Interrupt Structure**



## 10.2 Addressing Modes

The microcontroller core has seven instruction addressing modes: inherent, immediate, direct, indirect, indexed, absolute jump and relative jump (see Table 24).

### Inherent

The inherent addressing mode instructions either have no operand associated or the contents of the operand are already known to the microcontroller core. The microcontroller core then inherently knows how to execute the instruction without needing any additional information provided by additional operands.

### Immediate

The immediate addressing mode instructions contain a 3-bit,[2] 8-bit or 12-bit[3] immediate field as an operand. Immediate addressing is so-named because the value needed to complete the instruction is provided immediately to the core within the instruction code. That is to say, the instruction itself dictates what the data value is to be e.g. stored in a register.

### Direct

The direct addressing mode instructions contain an 8-bit address operand that directly points to a location within the data memory space. Direct addressing is so-named because the value needed to complete the instruction must be directly accessed by the core from the memory address provided by the instruction code.

### Indirect

The indirect addressing mode instructions use the content in XLO, X[7:0], to address a specific location within the data memory space (0x00 – 0xFF).[4] Indirect addressing is so-named because the value needed to complete the instruction must be retrieved indirectly by the core from the address provided by the X-pointer.

### Indexed

The indexed offset addressing mode instructions add an 8-bit unsigned offset value to the X-pointer yielding a new effective address to select a specific location anywhere within the memory map (both program and data memory space, 0x000-0xFFF). Indexed addressing expands the functions of indirect addressing by providing the only means to access the data stored within the program memory space.

### Absolute

The absolute jump addressing mode instructions (e.g. JMP and JSR) replace the program counter with the value in the operand field. This allows jumping to any location within the program memory space.[5]

### Relative

The opcode instruction field for the relative jump addressing mode instruction, JP, is calculated from the distance to the absolute program memory location in the operand addressing the next instruction to be executed. The base opcode for JP is 0xC0 where bit 5 indicates the direction within memory to jump. Bits 4 to 0 indicate the number of bytes to jump where the maximum distance is 31 bytes. If bit 5 is zero, the address for the next instruction executed is determined by subtracting the lower 5 bits of the opcode (0xC1-0xDF) from the program counter; otherwise, the lower 5 bits of the opcode (0xE0-0xFF) are added to the program counter.[6]

**Table 24. Instruction Addressing Modes**

| Instruction | Immediate | | | Direct | | Indexed | Indirect | Inherent | | Relative | Absolute |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ADC | A, # | | | A, M | | A, [#, X] | A, [X] | | | | |
| ADD | A, # | | | A, M | | A, [#, X] | A, [X] | | | | |
| AND | A, # | | | A, M | | A, [#, X] | A, [X] | | | | |
| OR | A, # | | | A, M | | A, [#, X] | A, [X] | | | | |
| SUBC | A, # | | | A, M | | A, [#, X] | A, [X] | | | | |
| XOR | A, # | | | A, M | | A, [#, X] | A, [X] | | | | |
| CLR | | | | M | | | | A | X | | |
| INC | | | | M | | | | A | X | | |
| DEC | | | | M | | | | A | X | | |
| IFEQ | A, # | X, # | M, # | A, M | | A, [#, X] | A, [X] | | | | |
| IFGT | A, # | X, # | | A, M | | A, [#, X] | A, [X] | | | | |
| IFNE | A, # | X, # | M, # | A, M | | A, [#, X] | A, [X] | | | | |
| IFLT | | X, # | | | | | | | | | |
| SC | | | | | | | | no-op | | | |
| RC | | | | | | | | no-op | | | |
| IFC | | | | | | | | no-op | | | |
| IFNC | | | | | | | | no-op | | | |
| INVC | | | | | | | | no-op | | | |
| LDC | | | | #, M | | | | | | | |
| STC | | | | #, M | | | | | | | |
| RLC | | | | M | | | | A | | | |
| RRC | | | | M | | | | A | | | |
| LD | A, # | X, # | M, # | A, M | M, M | A, [#, X] | A, [X] | | | | |
| ST | | | | A, M | | A, [#, X] | A, [X] | | | | |
| NOP | | | | | | | | no-op | | | |
| IFBIT | #, A | | | #, M | | | #, [X] | | | | |
| IFNBIT | #, A | | | #, M | | | #, [X] | | | | |
| SBIT | | | | #, M | | | #, [X] | | | | |
| RBIT | | | | #, M | | | #, [X] | | | | |
| JP | | | | | | | | | | Rel. | |
| JSR | | | | | | [#, X] | | | | | M |
| JMP | | | | | | [#, X] | | | | | M |
| RET | | | | | | | | no-op | | | |
| RETI | | | | | | | | no-op | | | |
| INTR | | | | | | | | no-op | | | |

## Table 25. Instruction Cycles and Bytes

| Mnemonic | Operand | Bytes | Cycles | Flags affected |
|---|---|---|---|---|
| ADC | A, [X] | 1 | 1 | C,H,Z,N |
| ADC | A, [#,X] | 2 | 3 | C,H,Z,N |
| ADC | A, M | 2 | 2 | C,H,Z,N |
| ADC | A, # | 2 | 2 | C,H,Z,N |
| ADD | A, [X] | 1 | 1 | Z,N |
| ADD | A, [#,X] | 2 | 3 | Z,N |
| ADD | A, M | 2 | 2 | Z,N |
| ADD | A, # | 2 | 2 | Z,N |
| AND | A, [X] | 1 | 1 | Z,N |
| AND | A, [#,X] | 2 | 3 | Z,N |
| AND | A, M | 2 | 2 | Z,N |
| AND | A, # | 2 | 2 | Z,N |
| CLR | X | 1 | 1 | Z |
| CLR | A | 1 | 1 | C,H,Z,N |
| CLR | M | 2 | 1 | C,H,Z,N |
| DEC | X | 1 | 1 | Z |
| DEC | A | 1 | 1 | Z,N |
| DEC | M | 2 | 2 | Z,N |
| IFBIT | #, A | 1 | 1 | None |
| IFBIT | #, M | 2 | 2 | None |
| IFBIT | #, [X] | 1 | 1 | None |
| IFC | | 1 | 1 | None |
| IFEQ | A, [#, X] | 2 | 3 | None |
| IFEQ | A, [X] | 1 | 1 | None |
| IFEQ | A, # | 2 | 2 | None |
| IFEQ | A, M | 2 | 2 | None |
| IFEQ | M, # | 3 | 3 | None |
| IFEQ | X, # | 3 | 3 | None |
| IFGT | A, [#, X] | 2 | 3 | None |
| IFGT | A, [X] | 1 | 1 | None |
| IFGT | A, # | 2 | 2 | None |
| IFGT | A, M | 2 | 2 | None |
| IFGT | X, # | 3 | 3 | None |
| IFLT | X, # | 3 | 3 | None |
| IFNBIT | #, A | 1 | 1 | None |
| IFNBIT | #, M | 2 | 2 | None |
| IFNBIT | #, [X] | 1 | 1 | None |
| IFNC | | 1 | 1 | None |
| IFNE | A, [#, X] | 2 | 3 | None |
| IFNE | A, [X] | 1 | 1 | None |
| IFNE | A, # | 2 | 2 | None |
| IFNE | A, M | 2 | 2 | None |
| IFNE | X, # | 3 | 3 | None |
| IFNE | M, # | 3 | 3 | None |
| INC | A | 1 | 1 | Z,N |
| INC | M | 2 | 2 | Z,N |

| Mnemonic | Operand | Bytes | Cycles | Flags affected |
|---|---|---|---|---|
| INC | X | 1 | 1 | Z |
| INTR | | 1 | 5 | None |
| INVC | | 1 | 1 | C |
| JMP | M | 3 | 4 | None |
| JMP | [#, X] | 2 | 3 | None |
| JP | | 1 | 1 | None |
| JSR | M | 3 | 5 | None |
| JSR | [#, X] | 2 | 5 | None |
| LD | A, # | 2 | 2 | None |
| LD | A, [#,X] | 2 | 3 | None |
| LD | A, [X] | 1 | 1 | None |
| LD | A, M | 2 | 2 | None |
| LD | M, # | 3 | 3 | None |
| LD | M, M | 3 | 3 | None |
| LD | X, # | 3 | 3 | None |
| LDC | #, M | 2 | 2 | C |
| NOP | | 1 | 1 | None |
| OR | A, [X] | 1 | 1 | Z, N |
| OR | A, [#,X] | 2 | 3 | Z,N |
| OR | A, M | 2 | 2 | Z,N |
| OR | A, # | 2 | 2 | Z,N |
| RBIT | #, [X] | 1 | 2 | Z,N |
| RBIT | #, M | 2 | 2 | Z,N |
| RC | | 1 | 1 | C,H |
| RET | | 1 | 5 | None |
| RETI | | 1 | 5 | None |
| RLC | A | 1 | 1 | C,Z,N |
| RLC | M | 2 | 2 | C,Z,N |
| RRC | A | 1 | 1 | C,Z,N |
| RRC | M | 2 | 2 | C,Z,N |
| SBIT | #, [X] | 1 | 2 | Z,N |
| SBIT | #, M | 2 | 2 | Z,N |
| SC | | 1 | 1 | C,H |
| ST | A, [#,X] | 2 | 3 | None |
| ST | A, [X] | 1 | 1 | None |
| ST | A, M | 2 | 2 | None |
| STC | #, M | 2 | 2 | Z,N |
| SUBC | A, [X] | 1 | 1 | C,H,Z,N |
| SUBC | A, [#,X] | 2 | 3 | C,H,Z,N |
| SUBC | A, M | 2 | 2 | C,H,Z,N |
| SUBC | A, # | 2 | 2 | C,H,Z,N |
| XOR | A, [X] | 1 | 1 | Z,N |
| XOR | A, [#,X] | 2 | 3 | Z,N |
| XOR | A, M | 2 | 2 | Z,N |
| XOR | A, # | 2 | 2 | Z,N |

1. The FMS7401L's normal mode operation begins after a system reset and is when the 8-bit microcontroller core begins executing the instruction program residing in the code EEPROM memory. During this time, the code EEPROM memory may only be read by software not written. Refer to the Device Memory section of the datasheet for additional memory addressing information.
2. A 3-bit value in cases like the IFBIT and IFNBIT instructions.
3. A 12-bit value in the case of instructions writing to X.
4. The content of XHI (X[10:8]) is ignored.
5. The program memory space for the FMS7401L is 0xC00 to 0xFFF; however, the program counter will use only the 10 least significant bits of the address provided.
6. Although the JP instruction can jump forward 31 bytes, it can only jump backwards 30 bytes because the program counter is automatically incremented while the JP instruction is being executed.

# 11 Device Memory

The FMS7401L has 64 bytes of SRAM and 64 bytes of EEPROM (data EEPROM) available for data storage. It also has 1K Byte of EEPROM (code EEPROM) memory for program storage. During the device's normal operation, software has both read and write access of SRAM and data EEPROM memories but has only read access of the code EEPROM.[1] That is, the code EEPROM is protected from unauthorized writes that can corrupt its contents during normal operating conditions. The code EEPROM can only be written to when the device is in programming mode[2] and if the write disable (WDIS) bit of Initialization Register 1 is set to 0

While in normal operating mode, the user can write to the data EEPROM array by polling the ready (R) flag of the status register then executing the appropriate instruction. If the R flag is 1, the data EEPROM block is ready to perform the next write. If the R flag is 0, the data EEPROM is busy performing a write operation. The data EEPROM array will set the R flag to 1 after completing the write operation. Attempts to read, write, or enter Halt/Idle Mode while the data EEPROM is busy (R=0) can affect the current data being written and cause the intruding read or write command to also fail.

The SRAM, data EEPROM, code EEPROM, and all other data register are memory mapped for easy access by software (see Table 30). The microcontroller core has an 11-bit X-pointer register that may be used to address data bytes within the memory map.[3] Bit 10 of the X-pointer (X[10] or XHI[1]) selects between the code and data memory space within the memory map. When X[10] is set to 1, the X-pointer selects the code memory space (addr. 0xC00 to 0xFFF) physically addresses a byte in the code EEPROM memory. Since the code EEPROM memory is 1K bytes, it requires only 10 address bits to physically address a byte of its memory. Bits 9-0 of the X-pointer (X[9:0] or {XHI[1:0],XLO[7:0]}) is the physical address of the code EEPROM used during a byte read instruction operation. When X[10] is set to 0, the X-pointer automatically addresses the data memory space (addr. 0x00 to 0xFF). Bits 9-0 of the X-pointer is the memory mapped (not physical) address for the entire data memory space (including the SRAM, data EEPROM, and all other data registers) used during a byte read/write instruction operation. In addition, when using X-pointer instructions with the "[X]" syntax, only the lower 8 bits of X are considered addressing the data memory space only. However, instructions with the "[#0,X]" syntax allow read access of the code memory space for look-up tables, etc. When using the X-pointer to address a byte in either the data or code memory space, software should load X with its 12-bit memory mapped address.

## 11.1 Initialization Registers

The FMS7401L has four 8-bit wide non-volatile initialization registers that are only accessible by the user in programming mode (if the memory security bits are not enabled). Each register has a corresponding shadow volatile register that is automatically updated during a reset and is used to initialize specific on-chip peripherals.

The Initialization Register 1 contains the three memory security bits, three feature enable bits, and the clock selection bit. Table 26 provides a detailed description of the Initialization Register 1. This register is defaulted to zero by the factory.

The Initialization Register 2 contains the internal oscillator frequency trim setting, $F_{OSC}$.[4] Prior to leaving the factory, the internal oscillator is trimmed to the appropriate frequency and the non-volatile register is pre-programmed. During a reset, the volatile shadow register (at address 0xBA) is updated with the factory programmed trim value. The shadow register associated with the Initialization Register 2 is accessibly by software during normal operation and may be written to in order to perform fine adjustments e.g. of the PWM timer outputs. If the software saved the original factory trim value, the software may restore the frequency to its original frequency.[5]

The Initialization Register 3 contains the factory calibration values for the two internal analog comparator circuits (Brown-out Reset and Programmable Comparator). The calibration is performed in order to configure the comparators to their proper levels (see Table 27). The non-volatile register is preprogrammed prior to leaving the factory.

The Initialization Register 4 contains the factory calibration value for the internal current source generator as well as the default G0/T1HS1 and G5/T1HS2 port configuration. The factory calibrates the current source generator to ensure that, if enabled, G3 can source $I_{SRC}$[4] of current. During the initial clock cycles of the reset sequence, the shadow register is updated configuring the G0/T1HS1 and G5/T1HS2 I/O ports to their pre-determined initial states. This offers the capability of driving G0/T1HS1 and G5/T1HS2 high within the first $T_{DIO}$[4] after the device is powered. The non-volatile register is pre-programmed

prior to leaving the factory with the appropriate calibration value and with the ports configured as tri-state inputs. In programming mode, the default port configuration may be changed; however, be sure to maintain the factory current source calibration value since writes to a single register must affect all bits.

The Initialization Registers 1, 2, 3 and 4 can be read from and written to while in programming mode. However, re-trimming the internal oscillator and re-calibrating the analog circuits once the device has left the factory is discouraged and will void all device guarantees.

**Table 26. Initialization Register 1 Bit Definitions**

| Initialization Register 1 (volatile/non-volatile addr. 0xB9, 0xBB) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| CLK_ADJ | CMODE | unused | WDEN | BOREN | UBD | WDIS[6] | RDIS[6] |

| | | |
|---|---|---|
| (7) | CLKADJ | When set, the internal clock trimming register (volatile Initialization Register 2, Addr. 0xBA) can be accessed by the core in order to modify the internal clock frequency. |
| (6) | CMODE | Clock mode select: 0 = Internal Oscillator, 1 = External Oscillator |
| (4) | WDEN | If set, the on-chip processor Watchdog Timer resets are enabled. |
| (3) | BOREN | If set, the on-chip Brown-out Reset comparator circuit is enabled. |
| (2) | UBD | If set, write access of the upper 32 bytes of the data EEPROM (0x60-0x7F) is disabled in both programming and normal mode.[1,2] |
| (1) | WDIS[6] | If set, write access of the device memory is permanently disabled while in programming mode.[2] |
| (0) | RDIS[6] | If set, read access of the device memory is permanently disabled while in programming mode.[2] |

**Table 27. Initialization Register 3 Bit Definitions**

| Initialization Register 3 (volatile/non-volatile addr. 0xD0, 0xD1) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| unused | unused | BOR_TRIM | | | COMP_TRIM | | |

| | | |
|---|---|---|
| (5:3) | BOR_TRIM | These three bits allow for the calibration of the Brown-out Reset comparator circuit. |
| (2:0) | COMP_TRIM | These three bits allow for the calibration of the Programmable Comparator's upper range circuit. |

**Table 28. Initialization Register 4 Bit Definitions**

| Initialization Register 4 (volatile/non-volatile addr. 0xD3, 0xD4) | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Bit 7** | **Bit 6** | **Bit 5** | **Bit 4** | **Bit 3** | **Bit 2** | **Bit 1** | **Bit 0** |
| T1HS_DIR | T1HS2_LEV | T1HS1_LEV | ISOURCE_TRIM[4:0] | | | | |

| | | |
|---|---|---|
| (7) | T1HS_DIR | Initializes during reset, the T1HS1 (G0) and T1HS2 (G5) I/O ports both either inputs or outputs. This bit shadows directly to bits 0 and 5 of PORTGC. |
| (6:5) | T1HSx_LEV | Initializes during reset, the individual T1HS1 (G0) and T1HS2 (G5) I/O port level. These bits shadow directly to bits 0 and 5 of PORTGD. |
| (4:0) | ISOURCE_TRIM | These five bits allow for the calibration of internal current source generator. |

**Table 29. T1HS1 (G0) and T1HS2 (G5) Default Configuration**

| T1HS_DIR | T1HS2_LEV | T1HS1_LEV | Description |
|---|---|---|---|
| 0 | 0 | 0 | T1HS1 and T1HS2 tri-state inputs |
| 0 | 0 | 1 | T1HS1 input with pull-up, T1HS2 input tri-state |
| 0 | 1 | 0 | T1HS1 input tri-state, T1HS2 input with pull-up |
| 0 | 1 | 1 | T1HS1 and T1HS2 input with pull-up |
| 1 | 0 | 0 | T1HS1 and T1HS2 push-pull 0 outputs |
| 1 | 0 | 1 | T1HS1 push-pull 1 output, T1HS2 push-pull 0 output |
| 1 | 1 | 0 | T1HS1 push-pull 0 output, T1HS2 push-pull 1 output |
| 1 | 1 | 1 | T1HS1 and T1HS2 push-pull 1 outputs |

## 11.2 Memory Map

All I/O ports, peripheral registers, and core registers (except the accumulator and the program counter) are mapped into the memory space.

### Table 30. Memory Mapped Registers

| Address | Memory Space | Block | Contents |
|---|---|---|---|
| 0x00 – 0x3F | Data | SRAM | Data RAM |
| 0x40 – 0x7F | Data | EEPROM | Data EEPROM |
| 0x9D | Data | ADC | ADATA register[7] |
| 0x9F | Data | ADC | ADCNTRL1 register |
| 0xA0 | Data | ADC | ADCNTRL2 register |
| 0xA2 | Data | Prog. Comparator | DDELAY register |
| 0xA4 | Data | PWM Timer 1 | PSCALE register |
| 0xA5 | Data | PWM Timer 1 | DTIME register |
| 0xA6 | Data | PWM Timer 1 | T1CMPALO register |
| 0xA7 | Data | PWM Timer 1 | T1CMPAHI register |
| 0xA8 | Data | PWM Timer 1 | T1CMPBLO register |
| 0xA9 | Data | PWM Timer 1 | T1CMPBHI register |
| 0xAA | Data | PWM Timer 1 | T1RALO register |
| 0xAB | Data | PWM Timer 1 | T1RAHI register |
| 0xAC | Data | PWM Timer 1 | TMR1LO register[7] |
| 0xAD | Data | PWM Timer 1 | TMR1HI register[7] |
| 0xAE | Data | PWM Timer 1 | T1CNTRL register |
| 0xAF | Data | MIW | WKEDG register |
| 0xB0 | Data | MIW | WKPND register |
| 0xB1 | Data | MIW | WKEN register |
| 0xB2 | Data | I/O | PORTGD register |
| 0xB3 | Data | I/O | PORTGC register |
| 0xB4 | Data | I/O | PORTGP register[7] |
| 0xB5 | Data | Timer 0 | WDSVR |
| 0xB6 | Data | Timer 0 | T0CNTRL register |
| 0xB7 | Data | Clock | Halt Mode register |
| 0xB9 | Data | Init Register | Initialization Register 1 (volatile)[8] |
| 0xBA | Data | ACE Core | Internal Clock trimming register (volatile) |
| 0xBB | Data | Init Register | Initialization Register 1[8] |
| 0xBC | Data | Init Register | Initialization Register 2[8] |
| 0xBD | Data | Prog. Comparator | COMP register |
| 0xBE | Data | ACE Core | XHI register |
| 0xBF | Data | ACE Core | XLO register |
| 0xC0 | Data | ACE Core | Power mode clear (PMC) |
| 0xCE | Data | ACE Core | SP register |
| 0xCF | Data | ACE Core | Status Register (SR)[9] |
| 0xD0 | Data | ACE Core | Initialization Register 3 (volatile)[8] |
| 0xD1 | Data | Init Register | Initialization Register 3[8] |
| 0xD2 | Data | Signature | Device_ID register[7] |
| 0xD3 | Data | Init Register | Initialization Register 4 (volatile)[8] |
| 0xD4 | Data | Init Register | Initialization Register 4[8] |
| 0xC00 – 0xFF5 | Program | EEPROM | Code EEPROM |
| 0xFF6 – 0xFF7 | Program | ACE Core | Timer0 Interrupt vector |
| 0xFF8 – 0xFF9 | Program | ACE Core | PWM Timer1 Interrupt vector |
| 0xFFA – 0xFFB | Program | ACE Core | MIW Interrupt vector |
| 0xFFC – 0xFFD | Program | ACE Core | Software Interrupt vector |
| 0xFFE – 0xFFF | Program | ACE Core | ADC Interrupt vector |

## Table 31. Memory Mapped Registers and their Register Bit Definitions

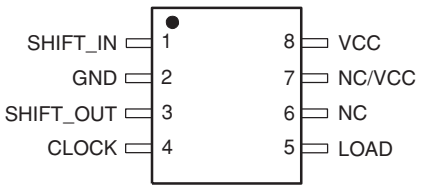
| Definitions of Register Bits | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Address** | **Name[10]** | **Bit7** | **Bit6** | **Bit5** | **Bit4** | **Bit3** | **Bit2** | **Bit1** | **Bit0** |
| 0x9D | ADATA | 8-bit digital value of ADC conversion | | | | | | | |
| 0x9F | ADCNTRL1 | APND | AINTEN | ASTART | REFSEL | ACHSEL [3:0] | | | |
| 0xA0 | ADCNTRL2 | REFBY2 | COMPSEL | ENAMP | ENDAS | ASPEED [1:0] | | ENIS | GAIN |
| 0xA2 | DDELAY | COMPEN | PWINT | EPWM | OFFMODE | DD [3:0] | | | |
| 0xA4 | PSCALE | PLLEN | FS [1:0] | | FSEL | FMODE | PS [2:0] | | |
| 0xA5 | DTIME | X | X | X | DT [4:0] | | | | |
| 0xA6 | T1CMPALO | Low 8 bits of 12-bit T1CMPA register | | | | | | | |
| 0xA7 | T1CMPAHI | X | X | X | X | High 4 bits of 12-bit T1CMPA register | | | |
| 0xA8 | T1CMPBLO | Low 8 bits of 12-bit T1CMPB register | | | | | | | |
| 0xA9 | T1CMPBHI | X | X | X | X | High 4 bits of 12-bit T1CMPB register | | | |
| 0xAA | T1RALO | Low 8 bits of 12-bit T1RA register | | | | | | | |
| 0xAB | T1RAHI | X | X | X | X | High 4 bits of 12-bit T1RA register | | | |
| 0xAC | TMR1LO | Low 8 bits of 12-bit TMR1 register | | | | | | | |
| 0xAD | TMR1HI | X | X | X | X | High 4 bits of 12-bit TMR1 register | | | |
| 0xAE | T1CNTRL | T1C3 | T1C2 | T1C1 | T1C0 | T1PND | T1EN | X | T1BOUT |
| 0xAF | WKEDG | Bit number = port number, Edge direction | | | | | | | |
| 0xB0 | WKPND | Bit number = port number, Pending flag for port | | | | | | | |
| 0xB1 | WKEN | Bit number = port number, Interrupt enable for port | | | | | | | |
| 0xB2 | PORTGD | Bit number = port number, Data when output, Weak pull-up when input | | | | | | | |
| 0xB3 | PORTGC | Bit number = port number, Input or output setting of port | | | | | | | |
| 0xB4 | PORTGP | Bit number = port number, Digital value at pin, Read-only | | | | | | | |
| 0xB5 | WDSVR | Accepts the value 0x1B as a watchdog service | | | | | | | |
| 0xB6 | T0CNTRL | WKINTEN | X | X | X | X | X | T0PND | T0INTEN |
| 0xB7 | HALT | Reserved | Reserved | Reserved | Reserved | Reserved | Reserved | EIDLE | EHALT |
| 0xBB | InitReg1 | CLK_ADJ | CMODE | unused | WDEN | BOREN | UBD | WDIS | RDIS |
| 0xBC | InitReg2 | 8-bit value used internally to trim the internal oscillator frequency | | | | | | | |
| 0xBD | COMP | CL [5:0] | | | | | | VLOOP | COUT |
| 0xBE | XHI | X | X | X | X | X | code/data | X [9:8] | |
| 0xBF | XLO | Low 8 bits of 11-bit X register | | | | | | | |
| 0xCE | SP | X | X | X | X | SP [3:0] | | | |
| 0xCF | STATUS | EE Ready | unused | unused | Global Int. | Zero | Carry | Half Carry | Negative |
| 0xD1 | InitReg3 | unused | unused | BOR_TRIM [2:0] | | | COMP_TRIM [2:0] | | |
| 0xD4 | InitReg4 | T1HS_DIR | T1HS2_LEV | T1HS1_LEV | ISOURCE_TRIM [4:0] | | | | |

1. The FMS7401L's normal mode operation begins after a system reset and is when the 8-bit microcontroller core begins executing the instruction program residing in the code EEPROM memory.
2. The FMS7401L must be placed in a special programming mode of operation in order to have full write and read access of all of the device memories. Refer to the In-circuit Programming Specification section of the datasheet for details.
3. Refer to the the 8-Bit Microcontroller Core section of the datasheet for additional details.
4. Refer to the Electrical Characteristics section of the datasheet.
5. The Initialization Register 2 shadow register will automatically be restored with its original factory setting during a system reset.
6. Once the read and/or write protection is enabled, the only possible external action of accessing the memory in programming mode is to issue a "Program Erase" command through the programming interface that clears the entire code EEPROM memory contents including the volatile Initialization Register 1. This allows full access to the user enabling new device memory programming for the single programming mode session (unless the non-volatile WDIS and RDIS bits are cleared). Refer to the In-circuit Programming Specification section of the datasheet for addition details.
7. The register can only be read.
8. The register cannot be access during normal operation only in programming mode.
9. All SR bits except for bit 7 (the global interrupt mask) are read only when using direct, indirect, or indexed instructions. Software cannot restore SR using the traditional microcontroller methods. Refer to the 8-Bit Microcontroller Core section of the datasheet for additional details.
10. A) Names in all capital letters are predefined in the assembler. B) Names of the individual bits are not predefined and must be definced using an EQU statement in the user program source code: for example, "APND EQU 7". C) The initialization registers listed are the non-volatile registers. Each register has a volatile shadow register.

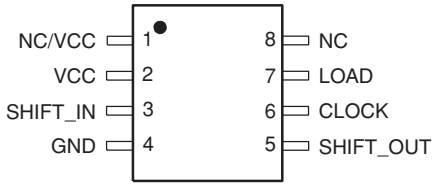
# 12 In-circuit Programming Specification

The FMS7401L supports in-circuit programming of all internal memory mapped registers including the data EEPROM, code EEPROM, and initialization registers. In-circuit programming consists of a 4-wire serial interface used to place the device in programming mode and issue all programming commands.[1] The external programmer should follow the device pinout[1] defined in Figure 18 and the timing rules defined by the parameters listed in Table 31 as shown in Figure 19 and Figure 20.

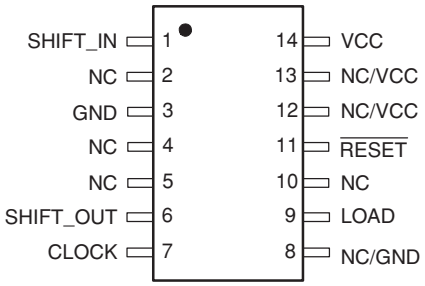
**Figure 18. Programming Mode Pin Configurations**




**FMS7401L 8-Pin PDIP/SOIC**




**FMS7401L 8-Pin TSSOP**




**FMS7401L 14-Pin PDIP/SOIC/TSSOP**

**Table 32. Programming Interface Electrical Characteristics[2]**

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $T_{HI}$ | CLOCK high time | 25 °C | 500 | | DC | nS |
| $T_{LO}$ | CLOCK low time | 25 °C | 500 | | DC | nS |
| $T_{DIS}$ | SHIFT_IN setup time | 25 °C | 100 | | | nS |
| $T_{DIH}$ | SHIFT_IN hold time | 25 °C | 100 | | | nS |
| $T_{DOS}$ | SHIFT_OUT setup time | 25 °C | 100 | | | nS |
| $T_{DOH}$ | SHIFT_OUT hold time | 25 °C | 900 | | | nS |
| $T_{ACCESS}$ | SHIFT_OUT sample time | 25 °C | 500 | | DC | nS |
| $T_{LOAD1}$, $T_{LOAD2}$, $T_{LOAD3}$, $T_{LOAD4}$ | Loading time | 25 °C | 5 | | | µS |
| $T_{READY}$ | EEPROM write time | 25 °C | | 3.7 | | mS |
| $T_{RESET}$ | System Reset time | 25 °C | | 3.7 | | mS |

## 12.1 Programming Mode Interface

In order to place the device in programming mode, a 10-bit opcode (0x34B) must be shifted into the device during its system reset. A system reset may be triggered during the device power-up by the Power-on Reset circuit or with the device already powered with a low pulse on the device RESET pin.[3] After power-up, the external programmer must shift in the 10-bit opcode before the system reset sequence completes. If the correct opcode is shifted, the device will automatically enter programming mode once the system reset sequence has completed.

The 10-bit opcode is serially shifted with the most significant bit (MSB) first into the device through the SHIFT_IN pin. Each opcode data bit must be valid by $T_{DIS}$ before the rising edge of CLOCK. As the opcode is shifted, the current 10-bit pattern is compared against 0x34B. If the 10-bit pattern is a match, the device will set the program mode flag and the device will enter programming mode once the system reset sequence completes (see Figure 19).

The opcode must be shifted in after Vcc settles to its nominal voltage level and before the system reset sequence ($T_{RESET}$) completes. Otherwise, the device will begin with its normal operation executing the instruction program residing in the code EEPROM memory. If an external reset is applied by bringing the RESET pin low, the 10-bit opcode may be shifted once RESET is released and before the system reset sequence completes.

## 12.2 Programming Protocol

Once the device is in programming mode, the programming protocol and commands may be issued. An externally controlled 4-wire interface consisting of a LOAD (G3) control, serial data SHIFT_IN (G4) input, serial data SHIFT_OUT (G2) output, and CLOCK (G1) pins are used to access the internal memory and registers. Communication between the external programmer and the FMS7401L is performed through a 32-bit command and response word, as described in Table 23. The serial data timing for the 4-wire interface is shown in Figure 20 and the programming protocol is shown in Figure 19. In order to exit programming mode, the device must be powered down or an external reset must be applied.

### 12.2.1 Byte Write Sequence

After the external programmer puts the FMS7401L into programming mode, the LOAD pin must be set to Vcc before serially shifting the first 32-bit command word using the SHIFT_IN and CLOCK signals. By definition, bit 31 of the command word must be shifted first followed by all other bits. With each bit of the 32-bit write command word shifted, the device shifts out a bit of the 32-bit response word from the previous command through the SHIFT_OUT pin. The external programmer may sample SHIFT_OUT after $T_{ACCESS}$ from the rising edge of CLOCK. The serial response word sent immediately after entering programming mode may contain indeterminate data.

After all 32 bits of the command word are shifted, the external programmer must set the LOAD signal to 0V and apply two clock pulses to the CLOCK signal, as shown in Figure 19, to complete the program cycle. Once the LOAD signal is brought low, the SHIFT_OUT pin acts as the handshaking signal between the device and external programmer hardware. When executing the write command, the device sets SHIFT_OUT low by the time the external programmer has issued the second rising edge of CLOCK informing the external programmer that the memory write is in progress. The external programmer must wait $T_{READY}$ for SHIFT_OUT to return high before returning the LOAD signal to Vcc to initiate the next command cycle.

### 12.2.2 Page Write Sequence

Page mode is a convenient and fast way to program the code EEPROM memory. In this mode, 16 bytes of data are written using a single write command followed by a stream of data bytes. Only full pages can be written in page mode where the address in the command word points to the beginning of a page.[4] After all 16 bytes of data has been shifted, the data will be written at once speeding up the total write time by a factor of 16 compared to byte mode programming. Figure 21 shows the page mode programming protocol.

Page mode's 32-bit write command word is similar to a byte write command except that bit 31 must be set to 1 in order to enable page mode. The address in the page-write command word (bits 17 to 8) must select the page to program the 16 bytes of data (the page address is a multiple of the page size: 0x000, 0x010, 0x020, etc.). The first byte of the page to program must be placed in the last 8 bits of the page-write command word (bits 7 to 0). All other bytes in the page must immediately follow after the initial page-write command has been entered.

The LOAD pin must be set to Vcc before serially shifting in the 32-bit page-write command word using the SHIFT_IN and CLOCK signals. By definition, bit 31 of the command word must be shifted first followed by all other bits. After all 32 bits of the command word are shifted, the external programmer must set the LOAD signal to 0V and apply two clock pulses to the CLOCK signal to latch the first byte of the page in its temporary data buffer. The LOAD signal must be returned to Vcc in order for the external programmer to shift the second byte of the page into the device (without repeating the command word). Once all 8 bits of the byte are shifted, the LOAD signal must again be set to 0V followed by two clocks pulses of the CLOCK signal in order to latch byte into its temporary data buffer. This process must be repeated until all 16 bytes are loaded into their data buffers.

While the $16^{th}$ byte of data is being latched, the actual write to the code EEPROM page, selected by the address in the 32-bit page-write command word, occurs. Once the LOAD signal is brought low, the SHIFT_OUT pin acts as the handshaking signal

between the device and external programmer hardware. The device sets SHIFT_OUT low during a page-write command by the time the external programmer has issued the second rising edge of CLOCK informing the programmer that the memory write is in progress. The external programmer must wait $T_{READY}$ for SHIFT_OUT to return high before returning the LOAD signal to Vcc to initiate the next command cycle.

### 12.2.3 Byte Read Sequence

The external programmer can only perform memory reads a byte at a time. Before shifting each new command, the external programmer must set the LOAD signal to Vcc. By definition, bit 31 of the command word must be shifted first followed by all other bits. With each bit of the 32-bit read command word shifted, the device shifts out a bit of the 32-bit response word from the previous command through the SHIFT_OUT pin. The external programmer must sample SHIFT_OUT after $T_{ACCESS}$ from the rising edge of CLOCK. The serial response word sent immediately after entering programming mode may contain indeterminate data.

After all 32 bits of the read command word are shifted, the external programmer must set the LOAD signal to 0V and apply two clock pulses to the CLOCK signal, as shown in Figure 19, to complete the read cycle. At the rising edge of the second clock pulse, the data read from the address provided in the read command word is latched into the lower 8-bits of its response word. Once LOAD is returned to Vcc, the next 32-bit command word may be shifted while the response word to the previous read command is shifted out with the data read from memory. If the last read command has been shifted, a dummy read command must be shifted to collect the last response word containing the last data byte read.

**Table 33. 32-Bit Command and Response Word**

| Bit Number | Input Command Word | Output Response Word |
|---|---|---|
| Bit 31 | Set to 1 to enable page mode memory access otherwise 0 for byte mode. | Same as the input command word. |
| Bit 30 | Must be set to 0. | Same as the input command word. |
| Bit 29 | Set to 1 to access the data memory space (data EEPROM or initialization registers) otherwise 0. | Same as the input command word. |
| Bit 28 | Set to 1 to access the code EEPROM otherwise 0. | Same as the input command word. |
| Bits 27 – 25 | Must be set to 0. | Same as the input command word. |
| Bit 24 | Set to 1 to perform a read or 0 to perform a write. | Same as the input command word. |
| Bit 22 | Set to 1 to perform a program memory erase otherwise 0. | Same as the input command word. |
| Bits 23, 21 – 18 | Must be set to 0. | Same as the input command word. |
| Bits 17 – 8 | Lower 10-bits of the memory mapped address byte to read/write or first byte of the page to write. | Same as the input command word. |
| Bits 7 – 0 | Data to be programmed if a write command or all zeros if a read command. | Same as the previous input write command word or the data read after an input read command word. |

**Figure 19. Programming Protocol**

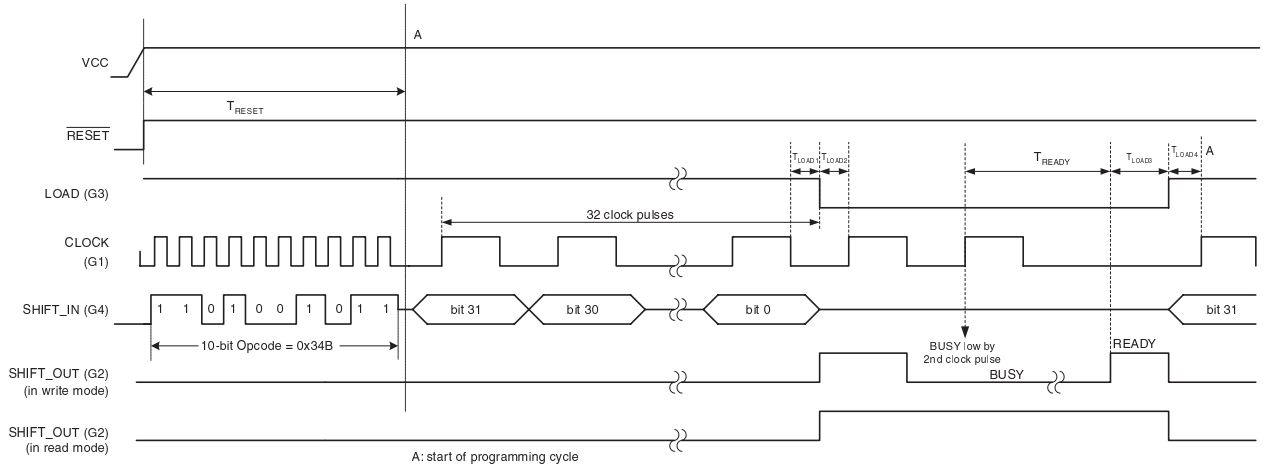



**Figure 20. Serial Data Timing**[1]

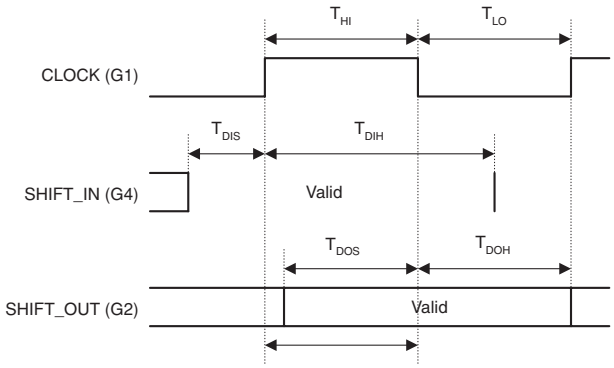
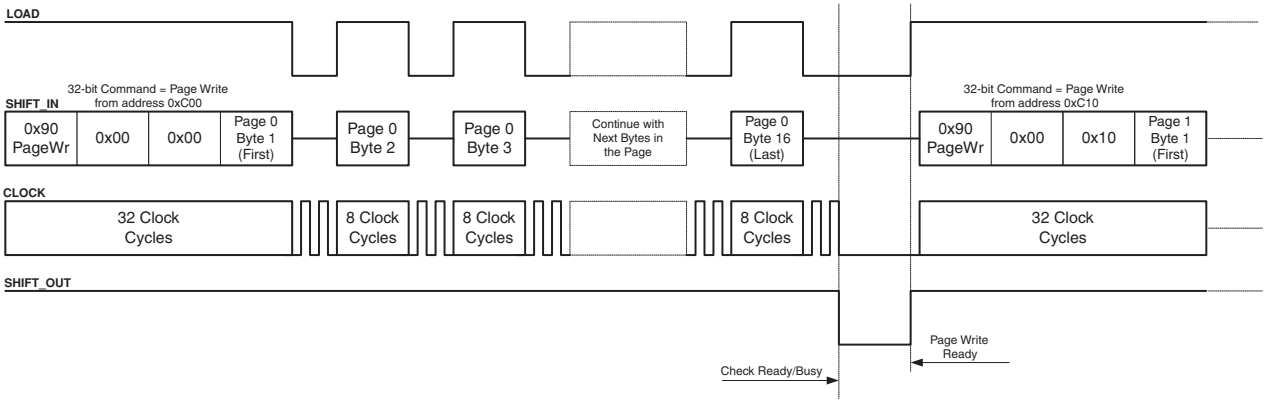



**Figure 21. Page Mode Protocol**



### 12.2.4 Program Memory Erase

The external programmer may erase the entire code EEPROM memory array using two special program erase byte write commands. This special erase option may also be used to unlock memory protected (WDIS/RDIS=1) devices without compromising design security. The special program erase byte write command overrides the WDIS memory security bit if set. Once both special byte write commands are issued, the volatile Initialization Register 1 is automatically cleared to unprotect the current programming mode session and allow complete access of the device memories.[5] The external programmer may then re-program the code EEPROM with a new pattern or permanently disable all security features by re-programming the non-volatile Initialization Register 1. All other memories, including the data EEPROM, are unaffected by the program erase commands.

The special program erase protocol requires the external programmer to shift two 32-bit command words addressing two separate page addresses. The special program erase 32-bit command word is similar to a byte write command except that bit 22 must be set to 1 to enable the program erase mode. The code EEPROM memory must also be selected by setting bit 28 of the command word. The first command word must select all even pages of the memory by setting the address bits (bits 17 to 8) to 0x000. The second command word must select all odd pages by setting the address bits to 0x010 of the command word. Any data value (bits 7 to 0) shifted as part of the individual command word may be used to erase the pages of the code EEPROM. After each even/odd page program erase command is executed, the even/odd pages of the code EEPROM memory is filled with the data supplied in the command erasing their previous program code data values. If the external programmer issues only one of the (even/odd page) erase commands, only half the pages will be erased by the data selected and the volatile Initialization Register 1 will not be cleared. Therefore, the current programming mode session will remain protected if either the memory protection (WDIS/RDIS) bits are set.

After the external programmer puts the FMS7401L into programming mode, the LOAD pin must be set to Vcc before serially shifting the first 32-bit program erase command word using the SHIFT_IN and CLOCK signals. By definition, bit 31 of the command word must be shifted first and then followed by all other bits. With each bit of the 32-bit write command word shifted, the device shifts out a bit of the 32-bit response word from the previous command through the SHIFT_OUT pin. The external programmer may sample SHIFT_OUT after $T_{ACCESS}$ from the rising edge of CLOCK. The serial response word sent immediately after entering programming mode may contain indeterminate data. After all 32 bits of the command word are shifted, the external programmer must set the LOAD signal to 0V and apply two clock pulses to the CLOCK signal, as shown in Figure 19, to complete the program cycle. Once the LOAD signal is brought low, the SHIFT_OUT pin acts as the handshaking signal between the device and external programmer hardware. When executing the write command, the device sets SHIFT_OUT low by the time the external programmer has issued the second rising edge of CLOCK informing the external programmer that the memory write is in progress. The external programmer must wait $T_{READY}$ for SHIFT_OUT to return high before returning the LOAD signal to Vcc to initiate the second program erase command cycle. The volatile Initialization Register 1 will only be cleared if both commands are successfully executed. All other memory accesses from this point forward are executed normally.

---

1. During in-circuit programming, G5 must be either not connected or driven high.
2. The following characteristics are guaranteed by design but are not 100% tested.
3. For addition detail regarding the device power-up and reset conditions refer the Reset Circuit section of the datasheet.
4. Each page in the code EEPROM has 16 bytes and starts at address 0xC00, 0xC10, 0xC20, etc.
5. For additional details regarding the WDIS, RDIS, and initialization registers, refer to the Device Memory section of the datasheet.

# 13 Electrical Characteristics

## Absolute Maximum Ratings

| Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|
| Ambient Storage Temperature | -65 | | +150 | °C |
| Input Voltage | -0.3 | | Vcc + 0.3 | V |
| Vcc Input Voltage | | 4.0 | | V |
| Lead Temperature (10s max) | | | +300 | °C |
| Electrostatic Discharge on all pins | 2000 | | | V |
| Internal Voltage Regulator output current | | 5 | | mA |

## Operating Conditions

| | |
|---|---|
| Relative Humidity (non-condensing) | 95% |
| EEPROM write limits | See AC Electrical Characteristics |

## 13.1 FMS7401L (2.7V to 3.6V)

### DC Electrical Characteristics

All measurements are valid for $T_A$=+25°C unless otherwise stated.

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $I_{CC}$[1] | Active Supply Current (without data EEPROM writes in progress) | $F_{ICLK}=F_{OSC}$[2] -40°C to 125°C | | 0.75 | 1.2 | mA |
| | Active Supply Current (with data EEPROM writes in progress) | $F_{ICLK}=F_{OSC}$[2] -40°C to 125°C | | 0.9 | 2.0 | mA |
| | Active Supply Current (without data EEPROM writes in progress) | $F_{ICLK}=F_{(FS=0)}$[2] -40°C to 125°C | | 6.5 | 13 | mA |
| $I_{STR}$[3] | Start-up Current | | | | 250 | µA |
| $I_{CCH}$ | Halt Mode Current | -40°C to 85°C | | 1.3 | 5 | µA |
| | | -40°C to 125°C | | 8.3 | 15 | µA |
| $I_{CCL}$[4] | Idle Mode Current | -40°C to 125°C | | 180 | 270 | µA |
| $S_{Vcc}$[3] | Power Supply Ramp Rate | | 1V/10mS | | 1V/1µs | |
| $V_{IL}$ | Input Low with Schmitt Trigger buffer | -40°C to 125°C | | | 0.2Vcc | V |
| $V_{IH}$ | Input High with Schmitt Trigger buffer | -40°C to 125°C | 0.8Vcc | | | V |
| $I_{TL}$ | Tri-State Leakage | -40°C to 125°C | | 0.01 | 1 | µA |
| $I_{IP}$ | Input Pull-up Current | $V_{IN}$=0V | | 85 | 350 | µA |
| $V_{OL}$ | Output Low Voltage (G1, G2, G3, G4, G6, G7) | 5mA sink current -40°C to 125°C | | | 0.3Vcc | V |
| | Output Low Voltage (G0, G5) | 2mA sink current -40°C to 125°C | | | 0.3Vcc | V |
| $V_{OH}$ | Output Low Voltage (G1, G2, G3, G4, G6, G7) | 5mA source current -40°C to 125°C | 0.7Vcc | | | V |
| | Output Low Voltage (G0, G5) | 2mA source current -40°C to 125°C | 0.7Vcc | | | V |

## AC Electrical Characteristics

All measurements are valid for $T_A$=+25°C unless otherwise stated.

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $F_{OSC}$[5] | Internal oscillator frequency (factory trim set-point) | Vcc=3.3V | 1.96 | 2.00 | 2.04 | MHz |
| Δckv | Internal oscillator frequency voltage variation | | -0.5 | | +0.5 | % |
| Δckt | Internal oscillator frequency temperature variation | Vcc=3.3V -40°C to 85°C | -3 | | +3 | % |
| | | Vcc=3.3V -40°C to 125°C | -4 | | +4 | % |
| $F_{PLL}$ | PLL input reference frequency | | | 2.00 | | MHz |
| $T_{PLL_LOCK}$[3] | PLL Lock time | -40°C to 125°C | | | 60 | µS |
| $T_{EEW}$ | EEPROM Writing time | | | 3.7 | 5 | mS |
| $T_{RESET}$[3] | System reset time | -40°C to 125°C | 2.5 | 3.7 | 4.7 | mS |
| $T_{DIO}$[3] | T1HS1 and T1HS2 default I/O configuration settling time | | | 40 | | µS |
| $T_{HALT_REC}$[3] | Internal device start time after exiting from Halt where $F_{ICLK} = F_{OSC}$[2] | -40°C to 125°C | | 5 | 7 | µS |

## Brown-out Reset (BOR) Electrical Characteristics

All measurements are valid for $T_A$=+25°C unless otherwise stated.

| Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| BOR Trigger Vcc Threshold Level | | 2.64 | 2.71 | 2.78 | V |
| | -40°C to +85°C | 2.60 | | 2.83 | V |
| | -40°C to 125°C | 2.59 | | 2.83 | V |

## Programmable Comparator Electrical Characteristics

All measurements are valid for $T_A$=+25°C unless otherwise stated.

| Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| All 32 thresholds ($V_{THU}$) | | -6% | $V_{THU}$ | +6% | V |
| Upper Range (0.45V to 2.0V) | -40°C to 125°C | -8% | $V_{THU}$ | +8% | V |
| All 31 thresholds ($V_{THL}$) | | $V_{THU}$ – 30mV | $V_{THL}$ | $V_{THU}$ + 30mV | V |
| Lower Range (0.03V to 0.43V) | -40°C to 125°C | $V_{THU}$ – 35mV | $V_{THL}$ | $V_{THU}$ + 35mV | V |
| Comparator Response Time[3] | 2mV overdrive | | 359 | | nS |
| | 5mV overdrive | | 173 | | nS |
| | 10mV overdrive | | 95 | | nS |

## ADC Electrical Characteristics

All measurements are valid for $T_A$=+25°C unless otherwise stated.

| Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| ADC Integral Non Linearity (INL) Best Fit[3] | $V_{AREF}$=Vcc<br>ASPEED=0 where ($F_{ICLK}$=$F_{OSC}$)/1[2] | | | 1.5 | LSB |
| | $V_{AREF}$=Internal Reference ($V_{REF}$)<br>ASPEED=0 where ($F_{ICLK}$=$F_{OSC}$)/1[2] | | | 1.5 | LSB |
| | $V_{AREF}$=Internal Reference ($V_{REF}$)<br>ASPEED=2 where ($F_{ICLK}$=$F_{OSC}$)/4[2]<br>-40°C to +125°C | | | 0.5 | LSB |
| ADC Differential Non Linearity (DNL)[3] | Vref=Vcc<br>ASPEED=0 where ($F_{ICLK}$=$F_{OSC}$)/1[2] | | | 2.5 | LSB |
| | $V_{AREF}$=Internal Reference ($V_{REF}$)<br>ASPEED=0 where ($F_{ICLK}$=$F_{OSC}$)/1[2] | | | 1.5 | LSB |
| | $V_{AREF}$=Internal Reference ($V_{REF}$)<br>ASPEED=2 where ($F_{ICLK}$=$F_{OSC}$)/4[2]<br>-40°C to +125°C | | | 1 | LSB |
| ADC Conversion Time[3] | ASPEED=0 where ($F_{ICLK}$=$F_{OSC}$)/1[2]<br>-40°C to +125°C | | | 20 | µS |
| Internal Voltage Reference ($V_{REF}$)[3] | | | 1.215 | | V |
| Amplifier x16 Gain Error[3] | -40°C to +125°C | 2 | | 2 | % |
| Current Source ($I_{SRC}$) on G3/AIN1 | | 0.9 | 1 | 1.1 | mA |
| Current Source ($I_{SRC}$) on G3/AIN1 | -40°C to +125°C | 0.89 | 1 | 1.11 | mA |

## Independent Amplifier Electrical Characteristics[3]

| Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|
| Input Bias Current | -40°C to +125°C | -1 | | +1 | µA |
| Input Offset Voltage | -40°C to +125°C | | 4 | | mV |
| Open Loop Voltage Gain | -40°C to +125°C | | 97 | | dB |
| Gain Bandwidth Product | -40°C to +125°C | | 3.7 | | MHz |
| Sink/Source Current | -40°C to +125°C | 0.5 | | 4.5 | mA |

**Figure 22. Internal Oscillator Frequency ($F_{OSC}$) vs. Temperature**

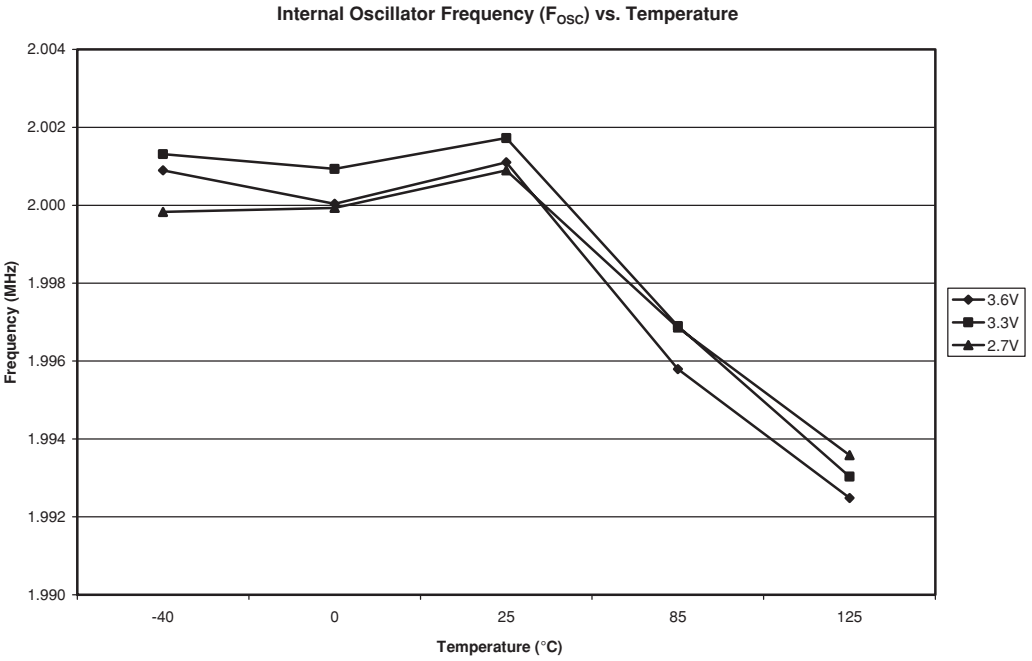



**Figure 23. Icc Active vs. Temperature (no PLL or data EEPROM writes)**

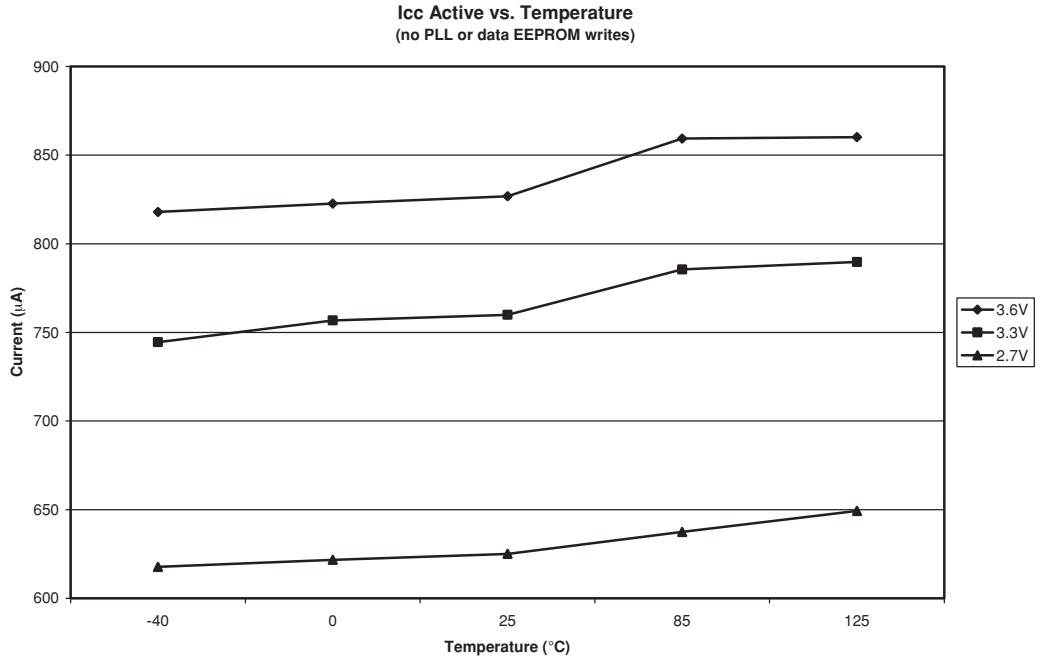


**Figure 24. Icc Active vs. Temperature (no PLL, with data EEPROM writes)**

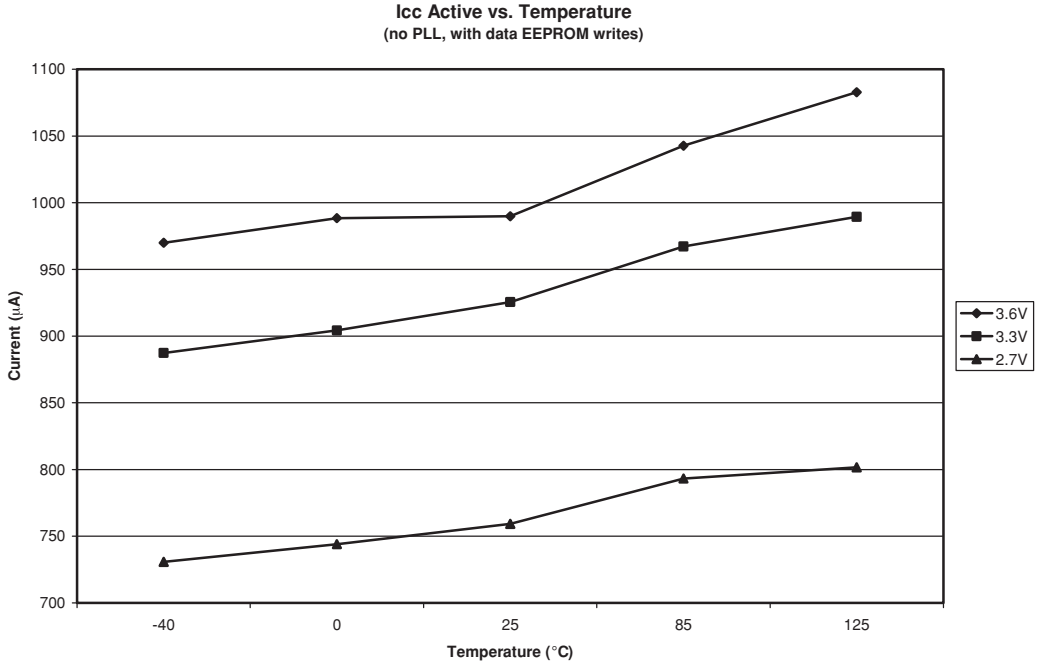



**Figure 25. Icc Active vs. Temperature (with PLL, no data EEPROM writes)**

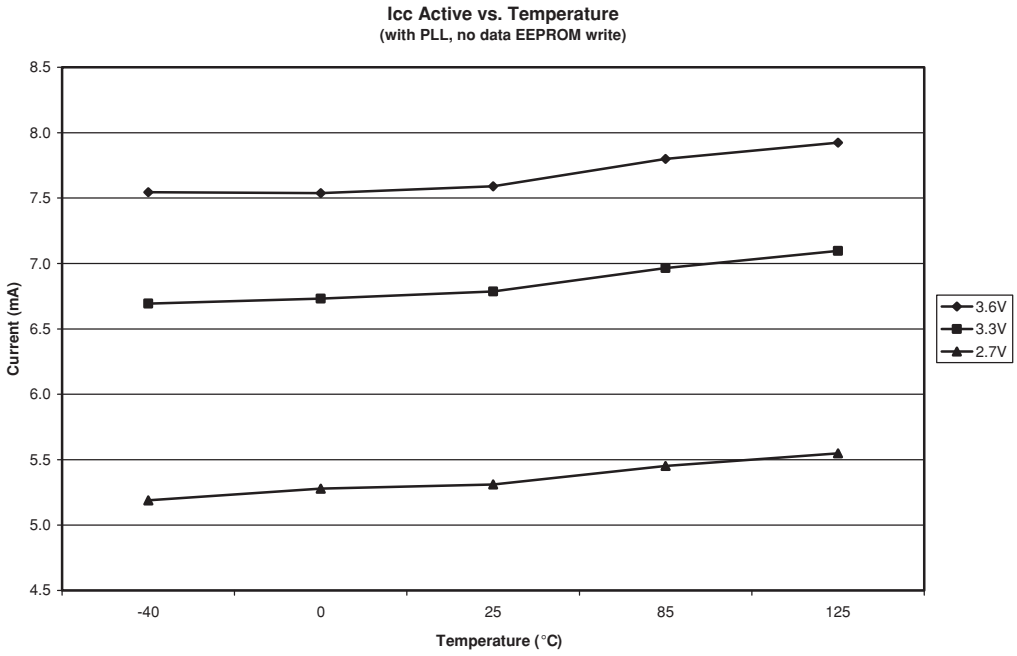


**Figure 26. Halt Current vs. Temperature**

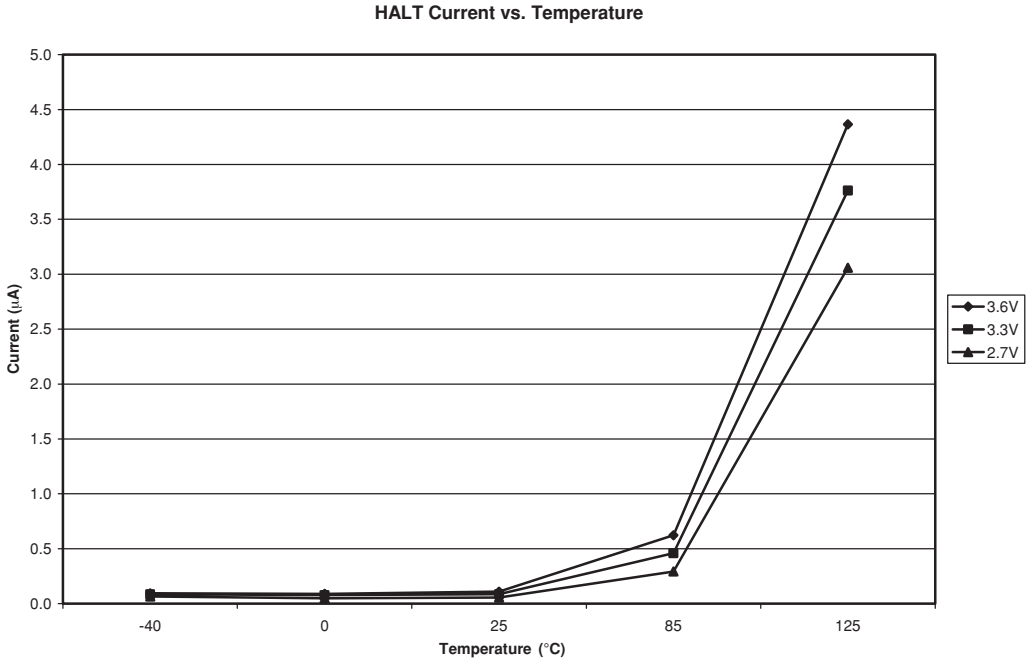



**Figure 27. Idle Current vs. Temperature (no PLL)**

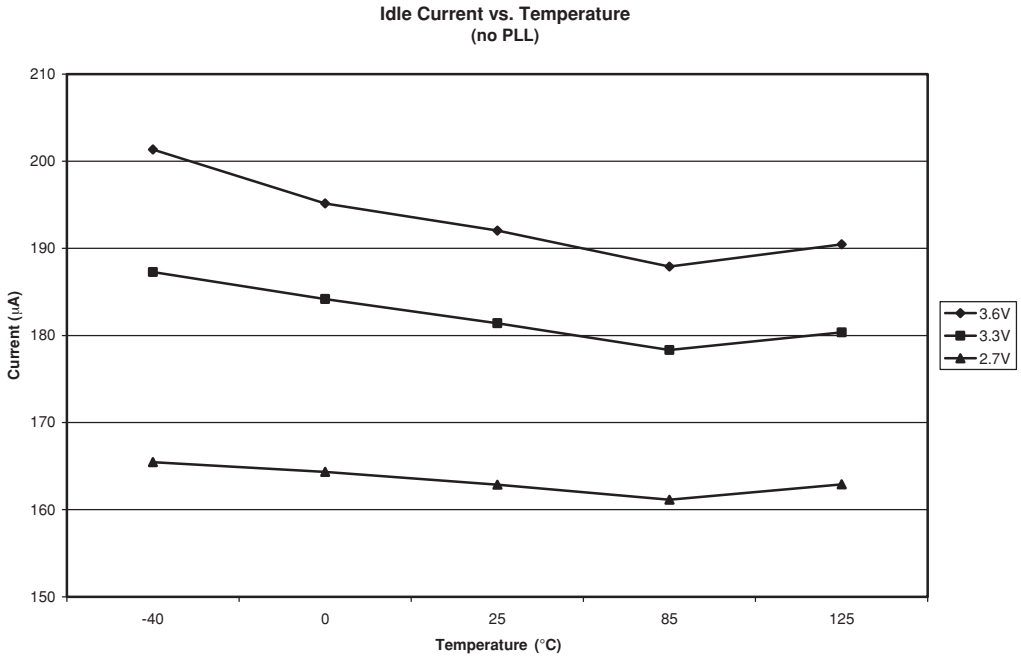


**Figure 28. Idle Current vs. Temperature (with PLL)**

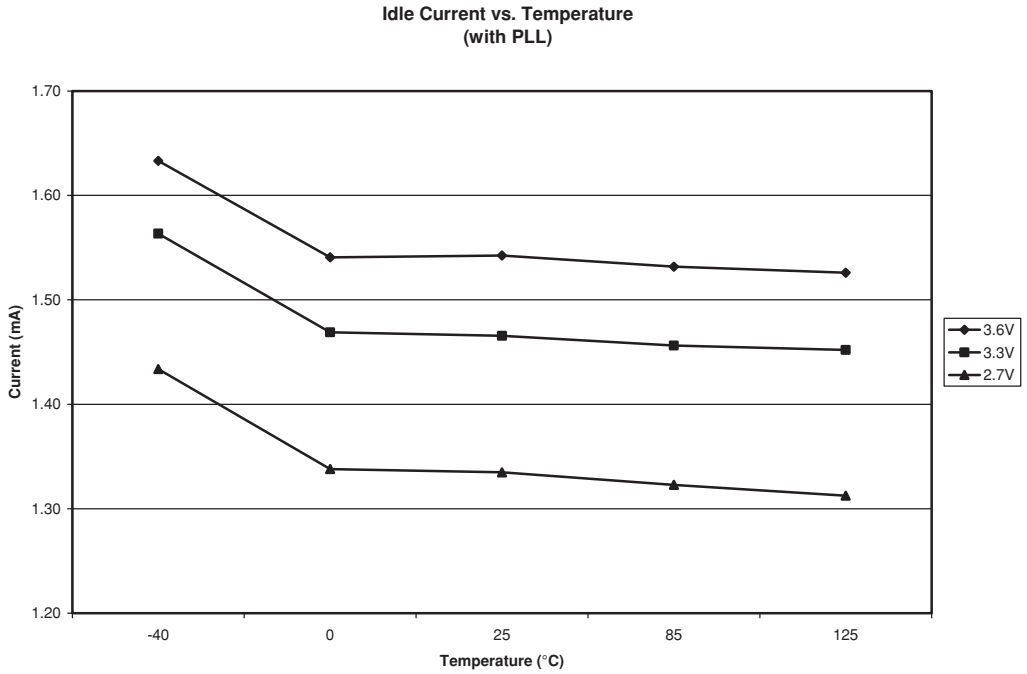



**Figure 29. $V_{OL}$ vs. $I_{OL}$ @ 25°C (G1–G4, G6, G7)**

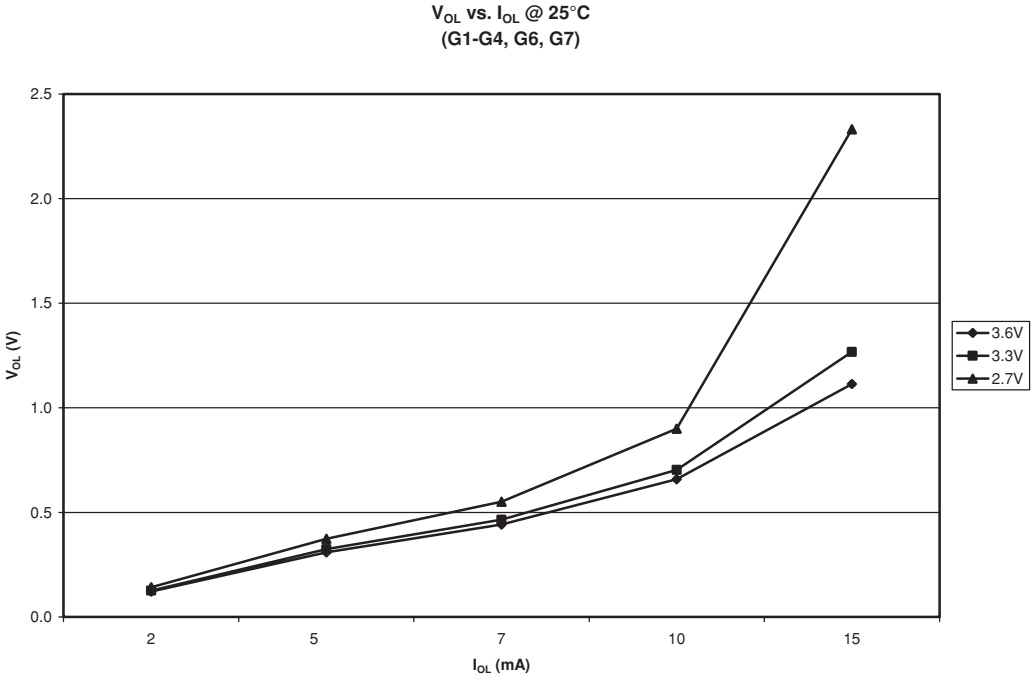


**Figure 30. $V_{OL}$ vs. $I_{OL}$ @ 25°C (G0, G5)**

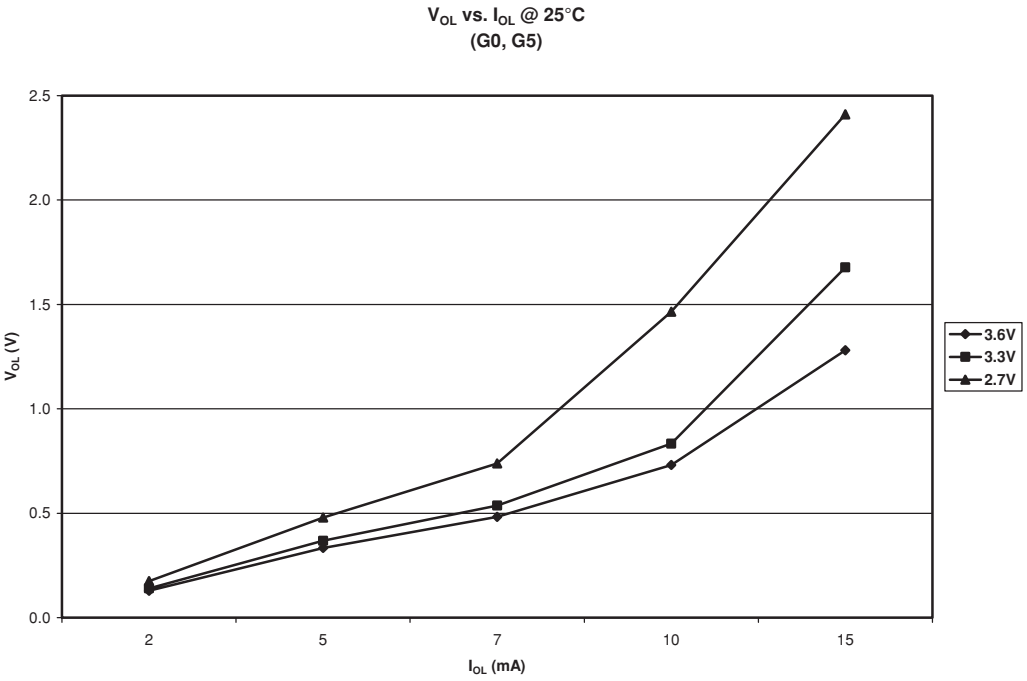



**Figure 31. $V_{OH}$ vs. $I_{OH}$ @ 25°C (G1–G4, G6, G7)**

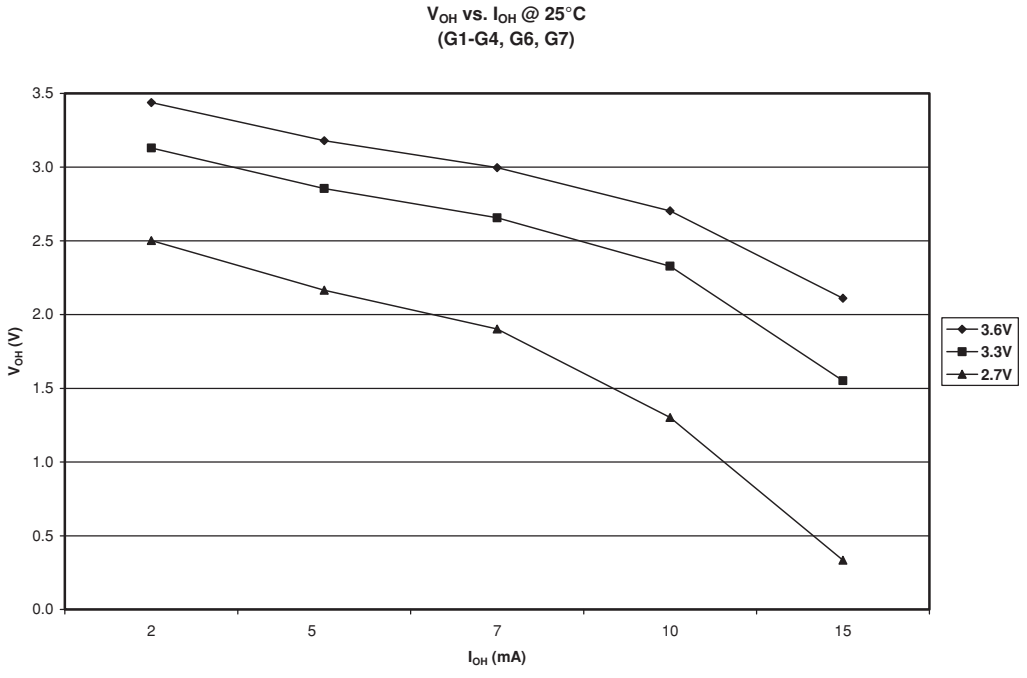


**Figure 32. $V_{OH}$ vs. $I_{OH}$ @ 25°C (G0, G5)**

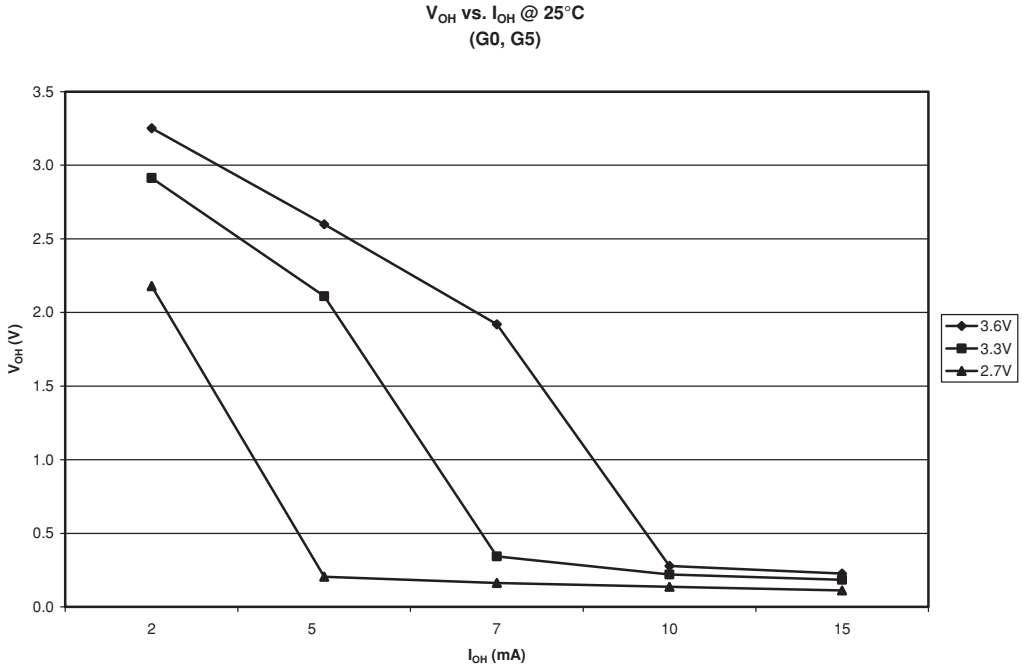



**Figure 33. BOR Level vs. Temperature**

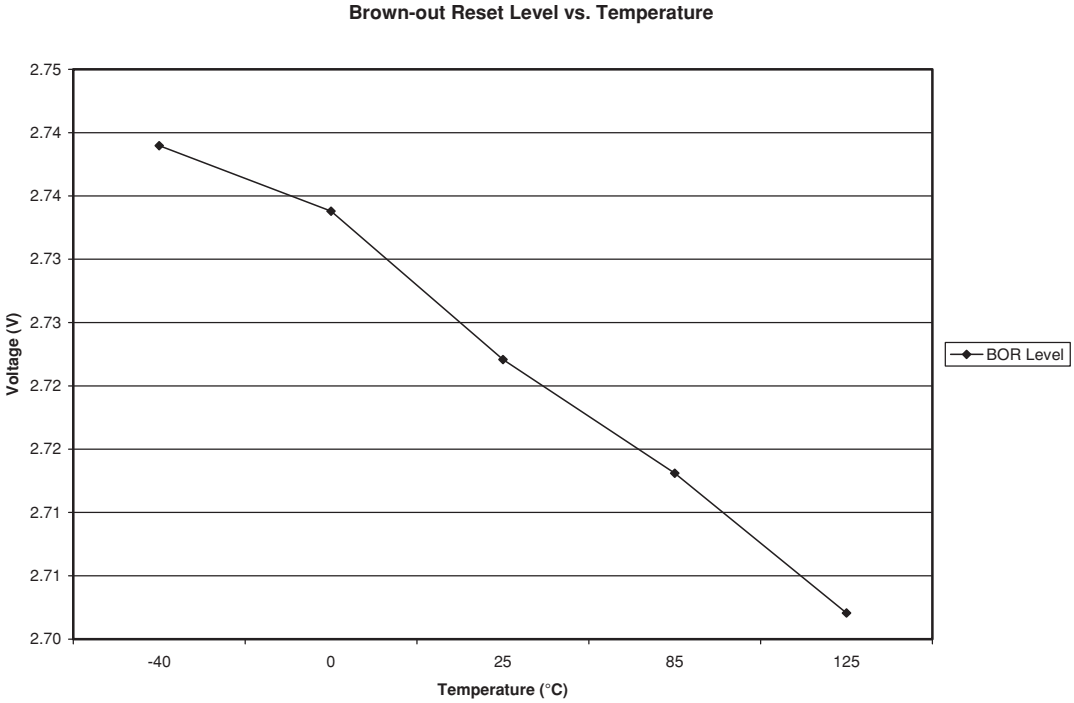


**Figure 34. Programmable Comparator Voltage Level vs. Temperature**

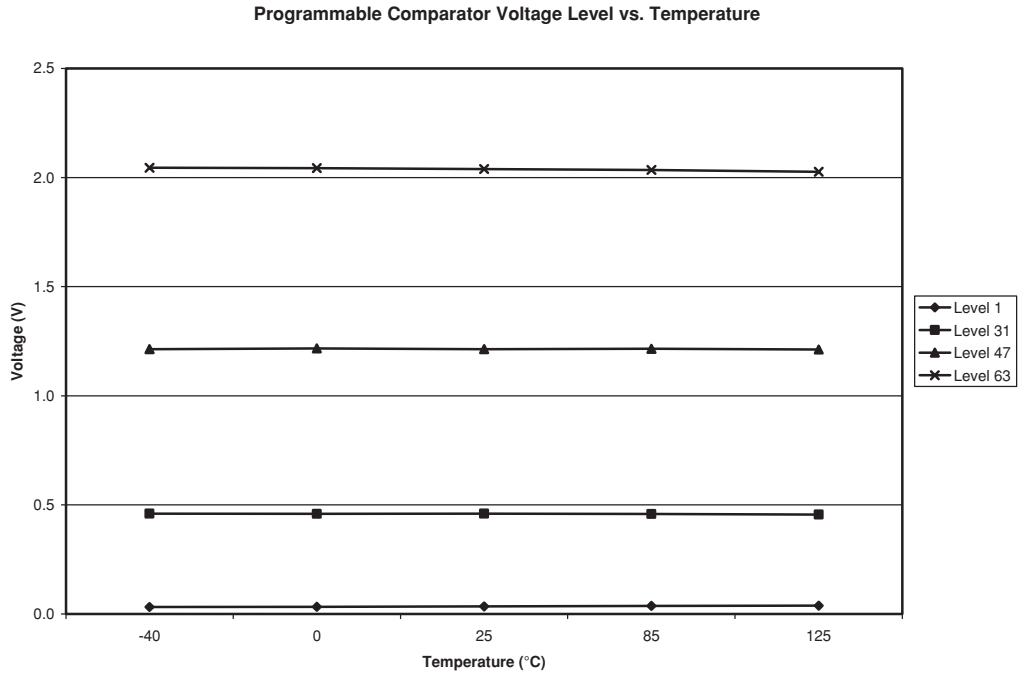



**Figure 35. $V_{REF}$ vs. Temperature**

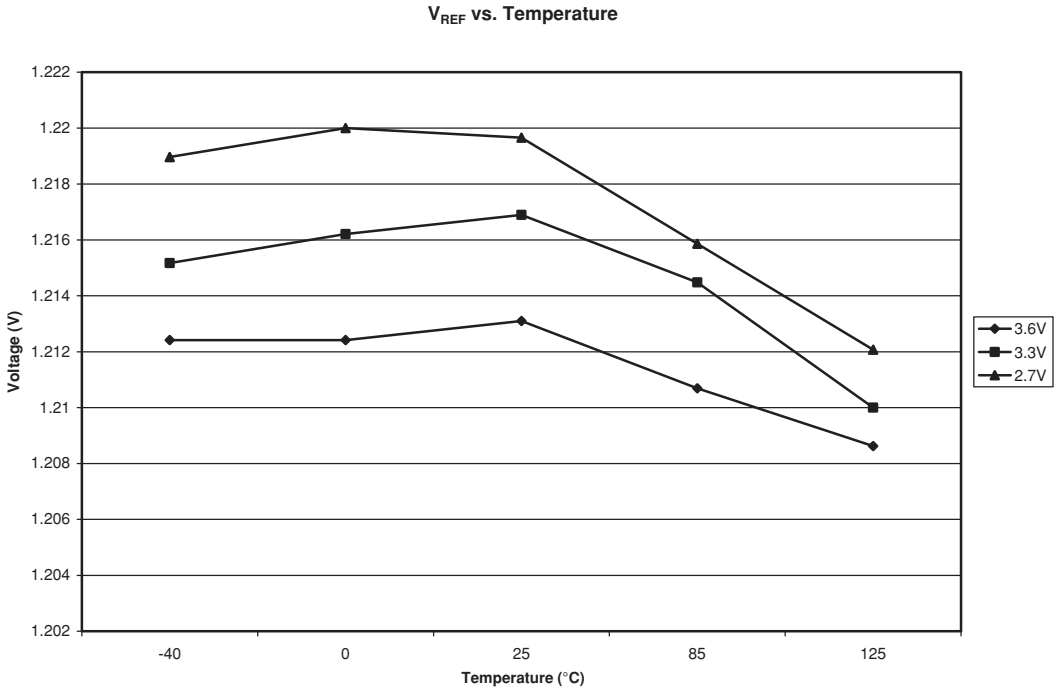


**Figure 36. Current Source ($I_{SRC}$) vs. Temperature**

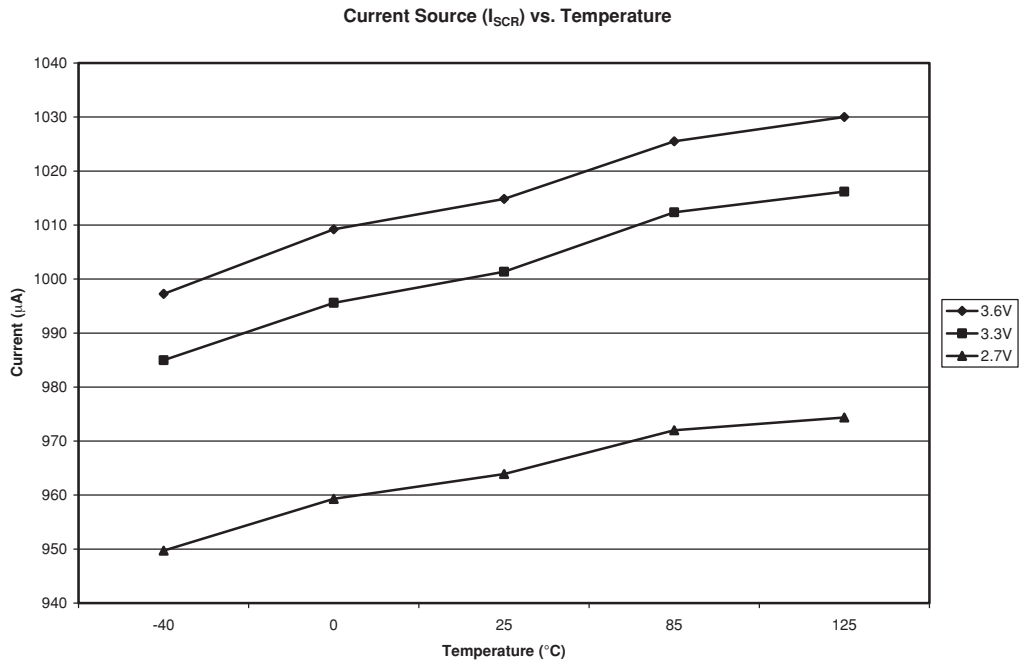
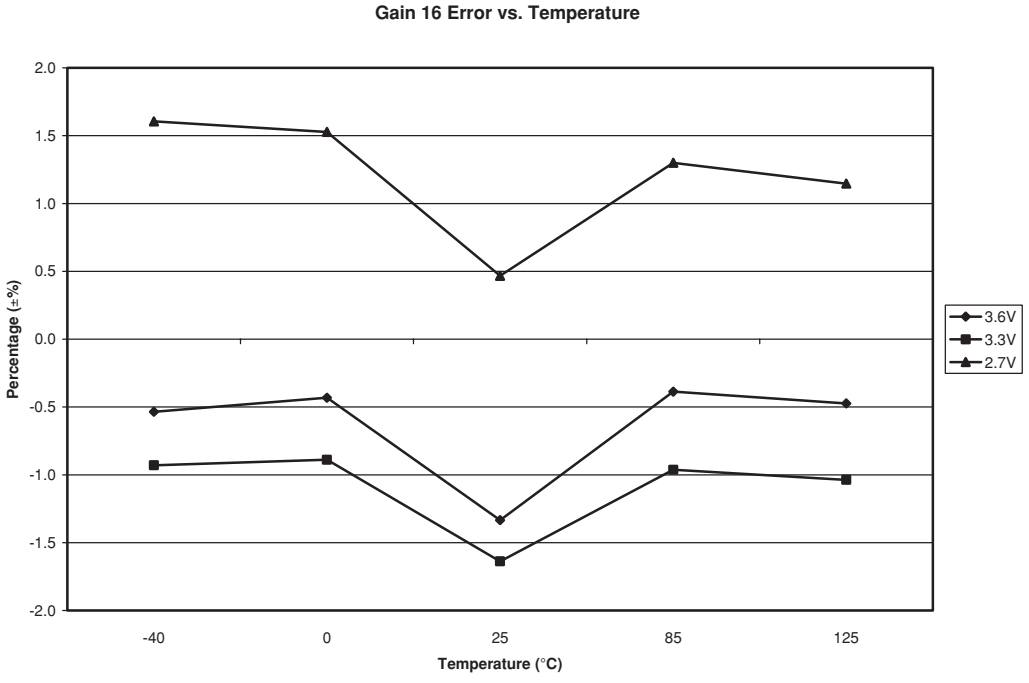



**Figure 37. Gain 16 Error vs. Temperature**



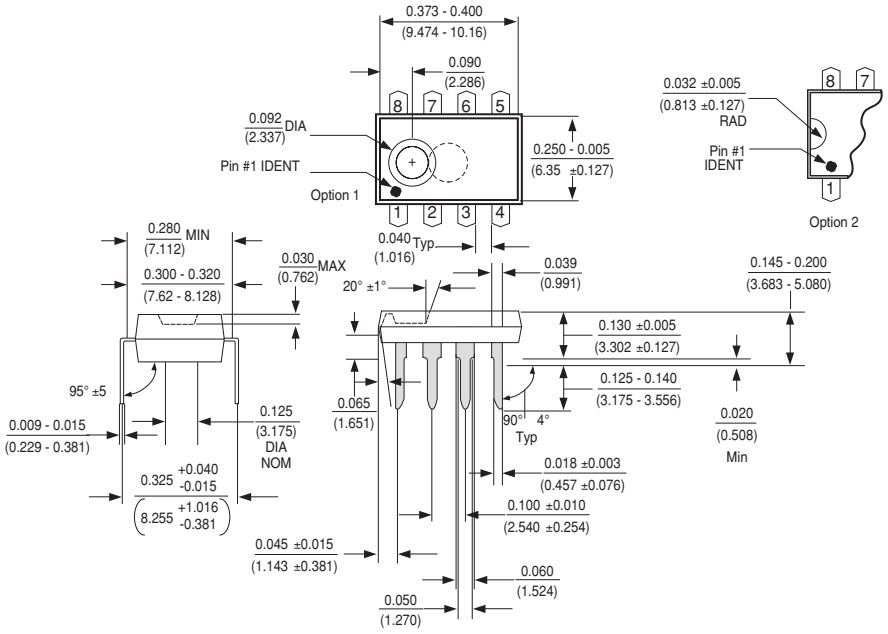
1. $I_{CC}$ Active current is dependent on the program code and number of active circuits. The $I_{CC}$ Active current specified is measured with the microcontroller core, PWM Timer 1, Timer 0, ADC, Uncommitted Amplifier, and Programmable Comparator circuits all active.
2. Refer to the Clock Circuit section of the datasheet for details regarding the FMS7401L's main system instruction clock ($F_{ICLK}$) and its clock sources ($F_{OSC}$ or $F_{(FS=0)}$).
3. The parameter is guaranteed by design but is not 100% tested.
4. The $I_{CC}$ Idle current is based on a continuous Idle Mode looping program and is dependent on the program code.
5. The upper $F_{OSC}$ frequency (4MHz) device option is available upon request.

# Ordering Information

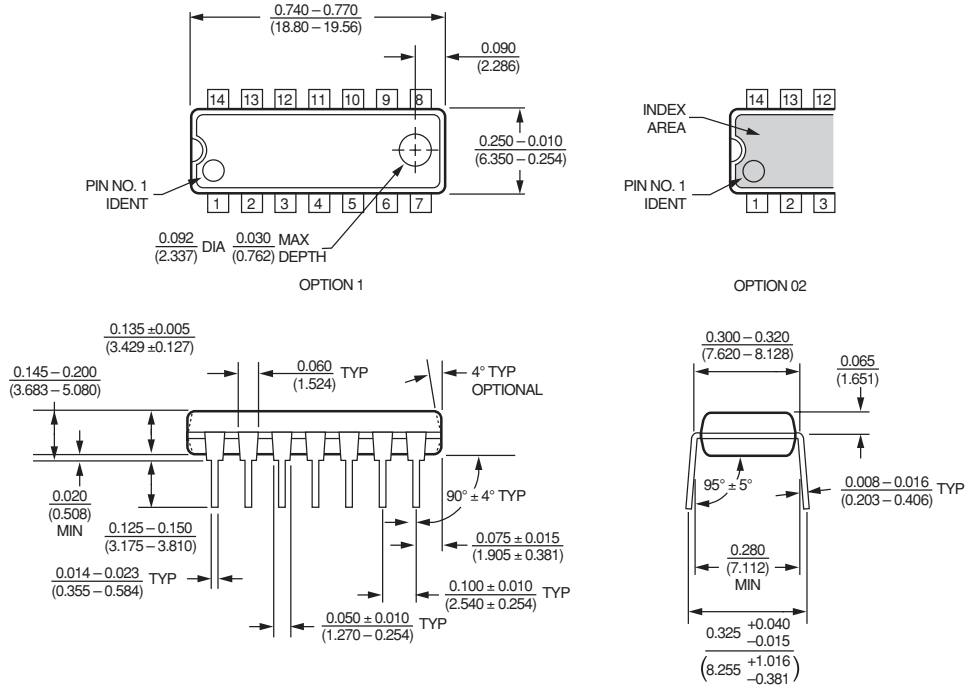
| FSID | Package | Supply Voltage | Temperature Range | Packaging Option | |
|---|---|---|---|---|---|
| | | | | Method | Qty |
| FMS7401LEN | PDIP8 | 2.7V to 3.6V | -40°C to 85°C | Rail | 40 |
| FMS7401LVN | PDIP8 | 2.7V to 3.6V | -40°C to 125°C | Rail | 40 |
| FMS7401LEN14 | PDIP14 | 2.7V to 3.6V | -40°C to 85°C | Rail | 25 |
| FMS7401LVN14 | PDIP14 | 2.7V to 3.6V | -40°C to 125°C | Rail | 25 |
| FMS7401LEM8X | SOIC8 | 2.7V to 3.6V | -40°C to 85°C | Tape Reel | 2500 |
| FMS7401LEMX | SOIC14 | 2.7V to 3.6V | -40°C to 85°C | Tape Reel | 2500 |
| FMS7401LEMT8X | TSSOP8 | 2.7V to 3.6V | -40°C to 85°C | Tape Reel | 2500 |
| FMS7401LEMTX | TSSOP14 | 2.7V to 3.6V | -40°C to 85°C | Tape Reel | 2500 |

# Physical Dimensions††

## 8-Pin PDIP



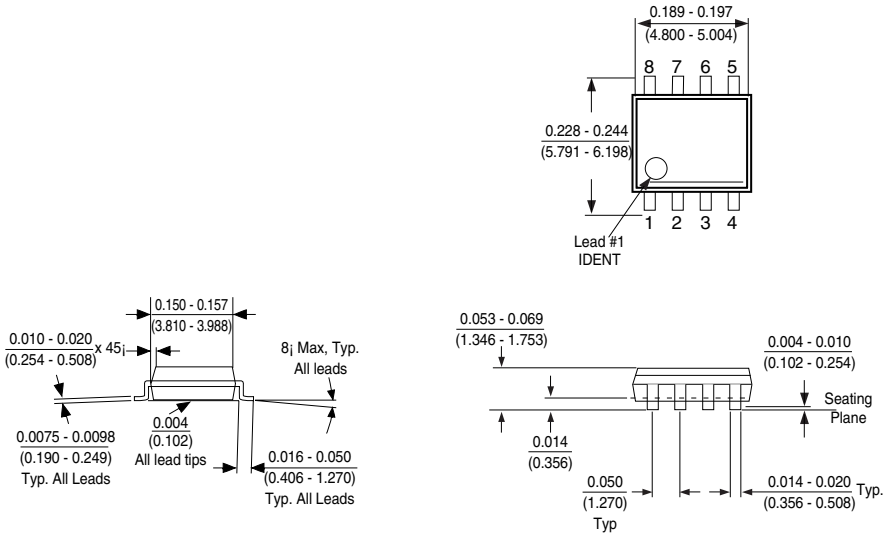

## 14-Pin PDIP



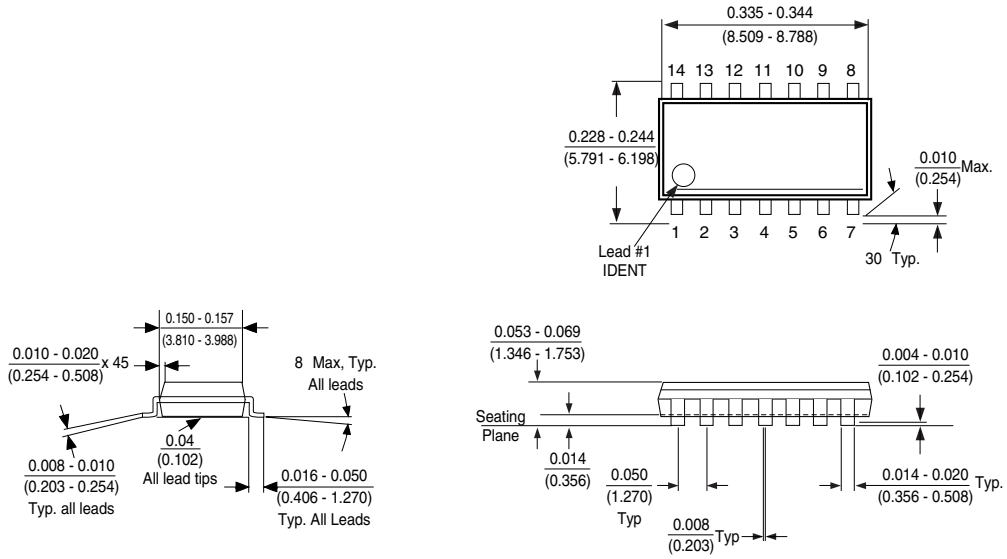

††Dimensions are in inches (millimeters) unless otherwise noted.

# Physical Dimensions††

## 8-Pin SOIC



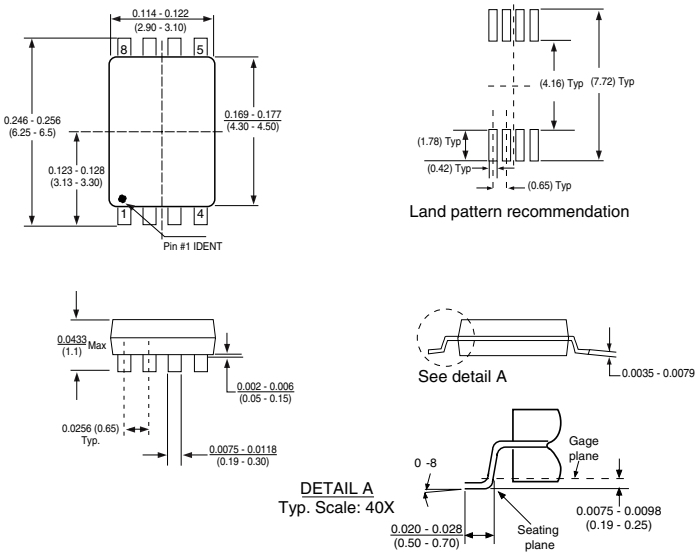

## 14-Pin SOIC




††Dimensions are in inches (millimeters) unless otherwise noted.

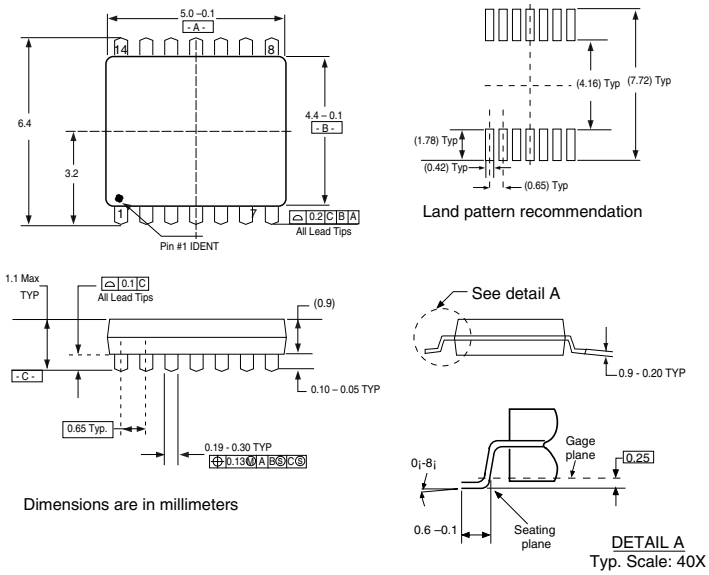
# Physical Dimensions††

## 8-Pin TSSOP




Notes: Unless otherwise specified
1. Reference JEDEC registration MO153. Variation AA. Dated 7/93

## 14-Pin TSSOP




Notes: Unless otherwise specified
1. Reference JEDED registration MO153. Variation AB.
Ref. Note 6, dated 7/93

††Dimensions are in inches (millimeters) unless otherwise noted.

**DISCLAIMER**

FAIRCHILD SEMICONDUCTOR RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN TO IMPROVE RELIABILITY, FUNCTION OR DESIGN. FAIRCHILD DOES NOT ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT DESCRIBED HEREIN; NEITHER DOES IT CONVEY ANY LICENSE UNDER ITS PATENT RIGHTS, NOR THE RIGHTS OF OTHERS.

**LIFE SUPPORT POLICY**

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury of the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.



1/24/05 0.0m 005
Stock#DS30007401
© 2004 Fairchild Semiconductor Corporation

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View FMS7401LEN14 on WIN SOURCE
- Fairchild/ON Semiconductor Information

## Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management