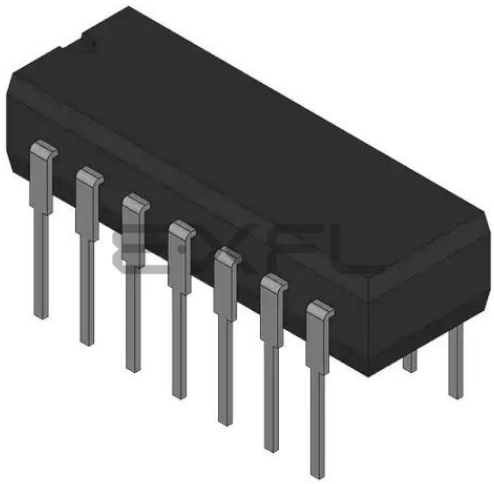


Welcome to E-XFL.COM

Embedded - Microcontrollers - Application Specific: Tailored Solutions for Precision and Performance

Embedded - Microcontrollers - Application Specific represents a category of microcontrollers designed with unique features and capabilities tailored to specific application needs. Unlike general-purpose microcontrollers, application-specific microcontrollers are optimized for particular tasks, offering enhanced performance, efficiency, and functionality to meet the demands of specialized applications.

What Are Embedded - Microcontrollers - Application Specific?

Application-specific microcontrollers are engineered to

Details	
Product Status	Obsolete
Applications	Digital Power Controller
Core Processor	8-Bit
Program Memory Type	EEPROM (1kB)
Controller Series	-
RAM Size	64 x 8
Interface	-
Number of I/O	8
Voltage - Supply	2.7V ~ 3.6V
Operating Temperature	-40°C ~ 125°C
Mounting Type	Through Hole
Package / Case	14-DIP (0.300", 7.62mm)
Supplier Device Package	14-DIP
Purchase URL	https://www.e-xfl.com/pro/item?MUrl=&PartUrl=fms7401lvn14

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.

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

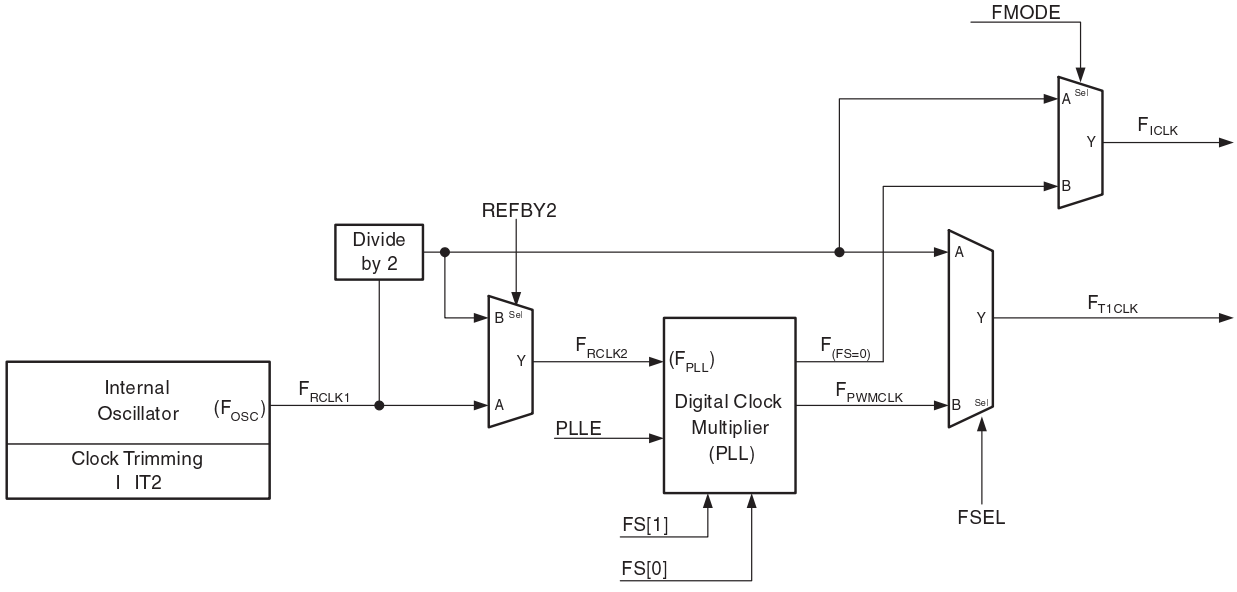
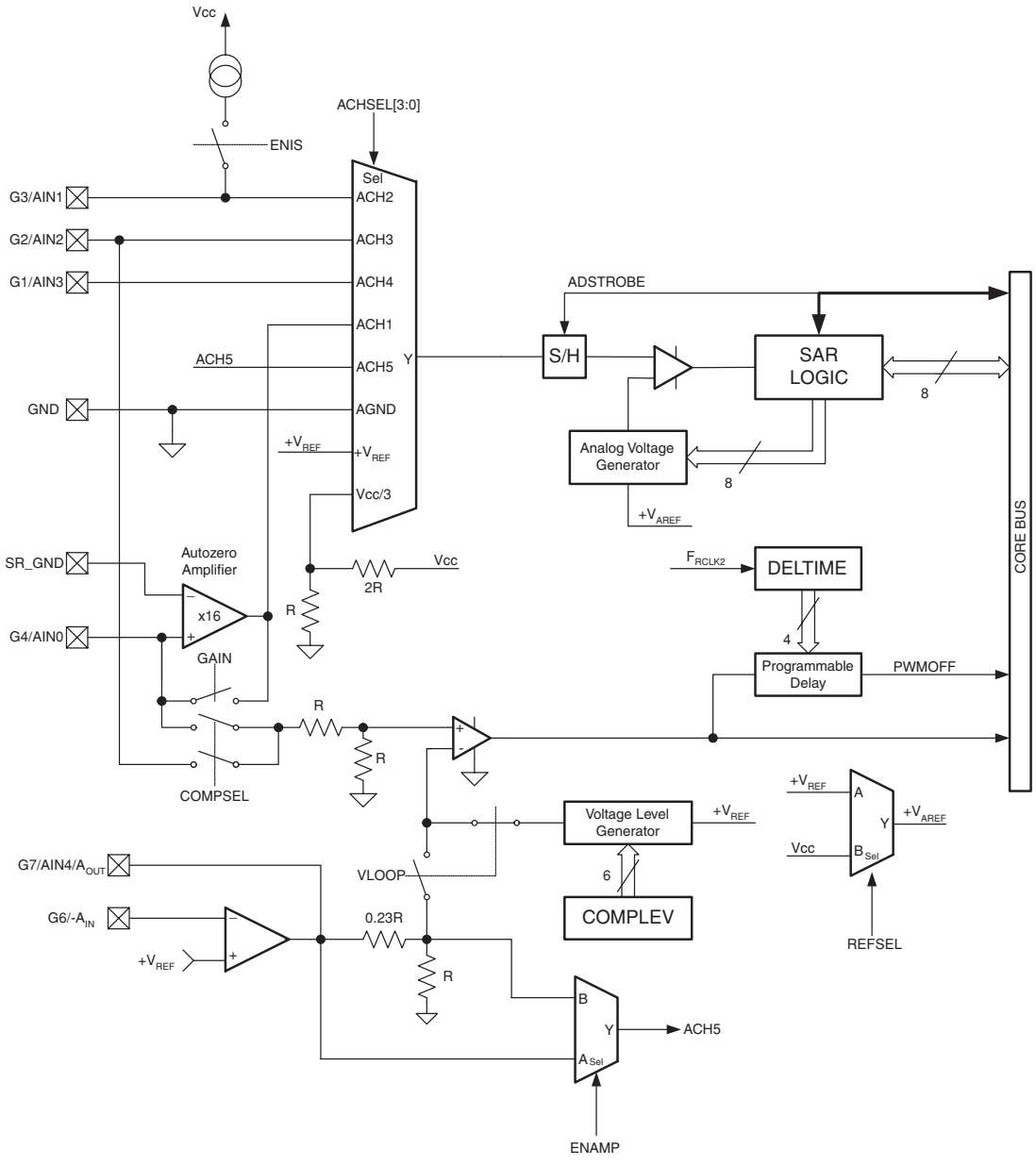


Figure 6. ADC Block Diagram[4]



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.

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.

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

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

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.

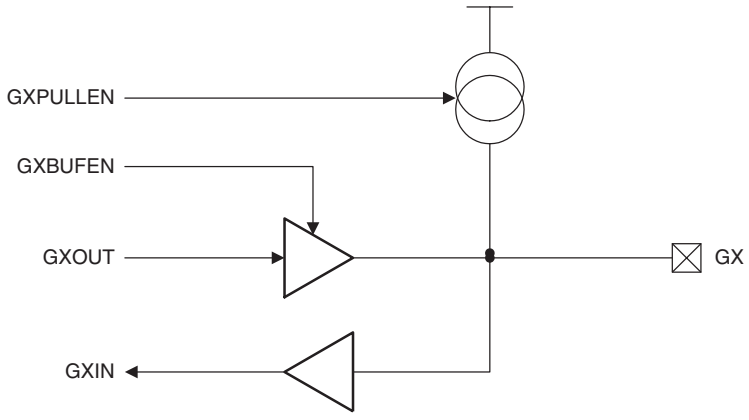
Bit 2 of the T1CNTRL register is the Timer 1's microcontroller hardware interrupt enable (T1EN) bit. If set, hardware interrupts are enabled and trigger by the T1PND and/or T1C0 pending flags depending on Timer 1's operating mode.[6] If in PWM Mode, the hardware interrupt is triggered only by the T1PND 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.

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.

10 8-Bit Microcontroller Core

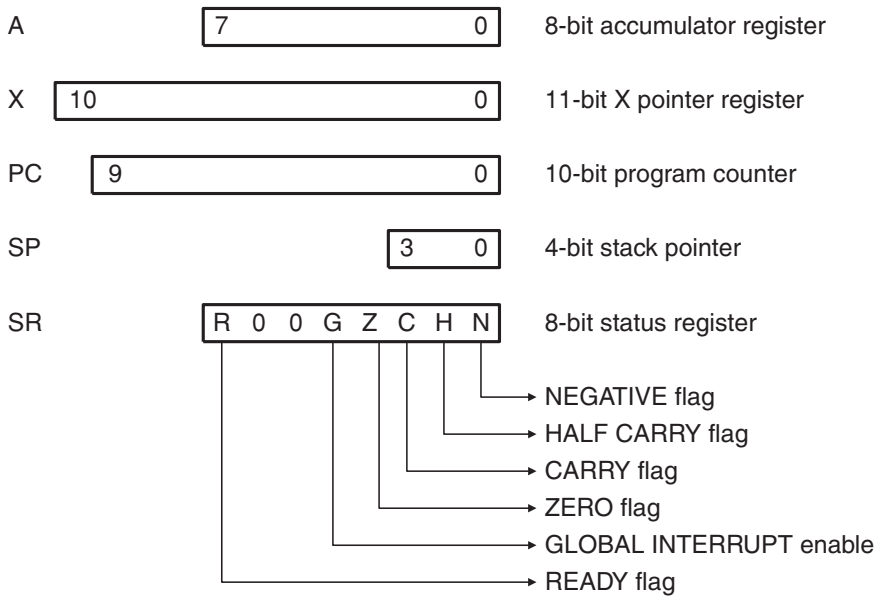
The FMS7401L's 8-bit microcontroller core is specifically designed for low cost applications involving bit manipulation, shifting and block encryption. It is based on a modified Harvard architecture meaning peripheral, I/O and RAM locations are addressed separately from instruction data.

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.

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			

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

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

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

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

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

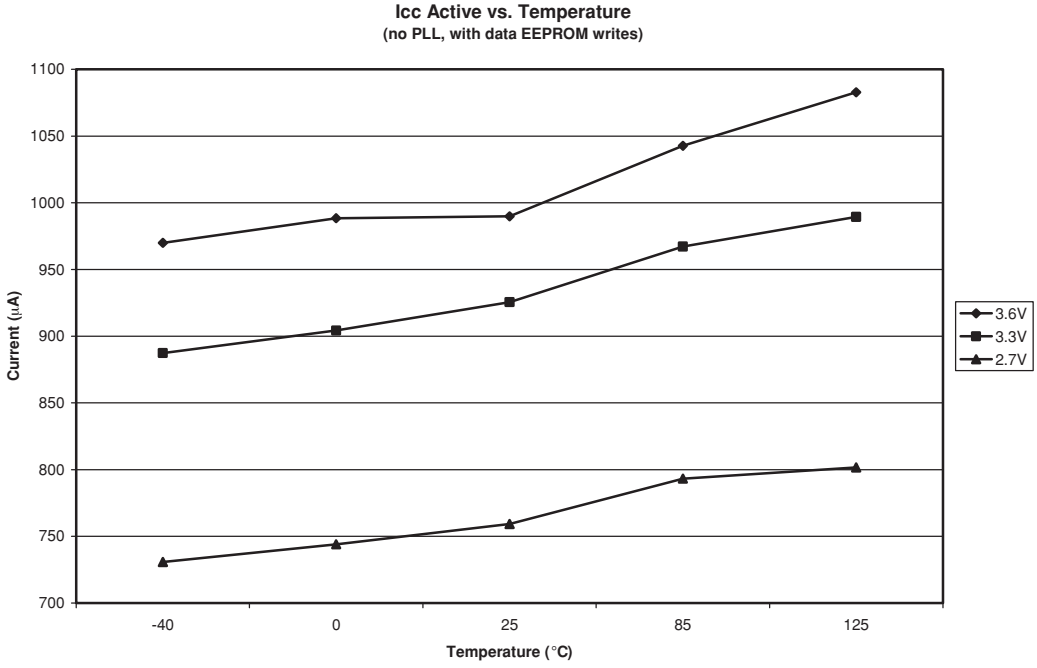
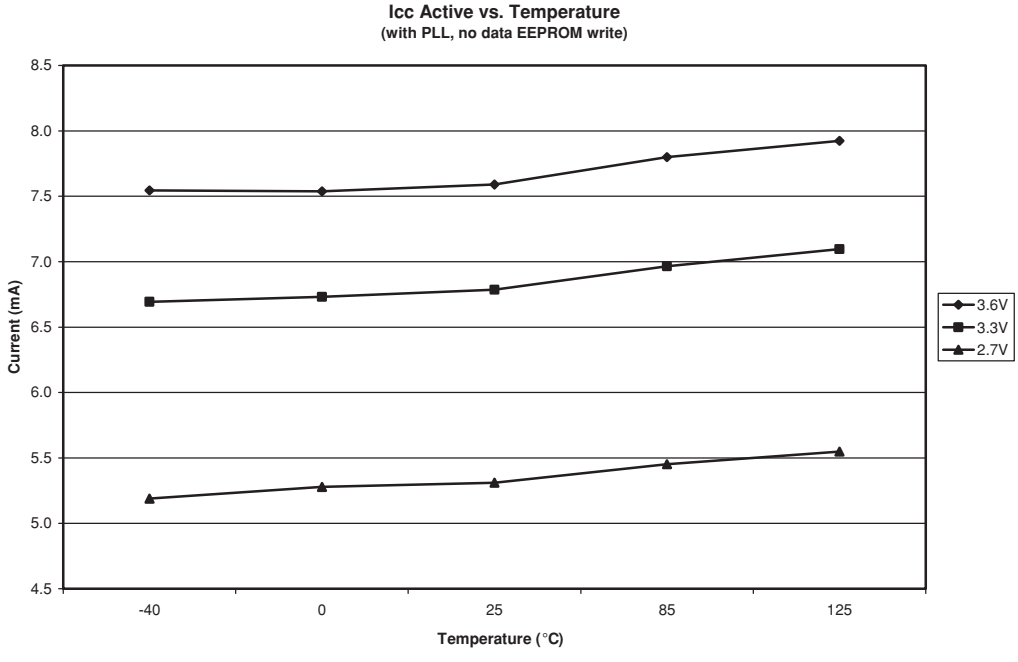


Figure 25. Icc Active vs. Temperature (with PLL, no data EEPROM writes)

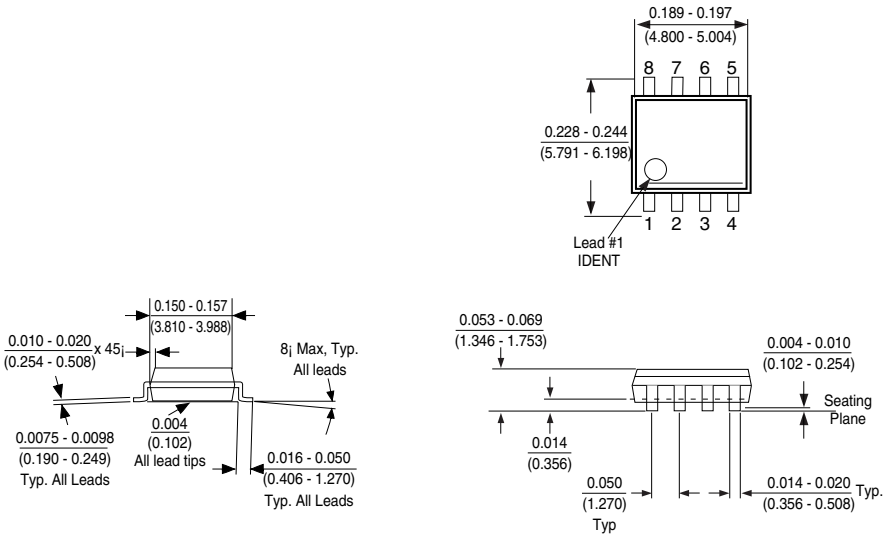


Ordering Information

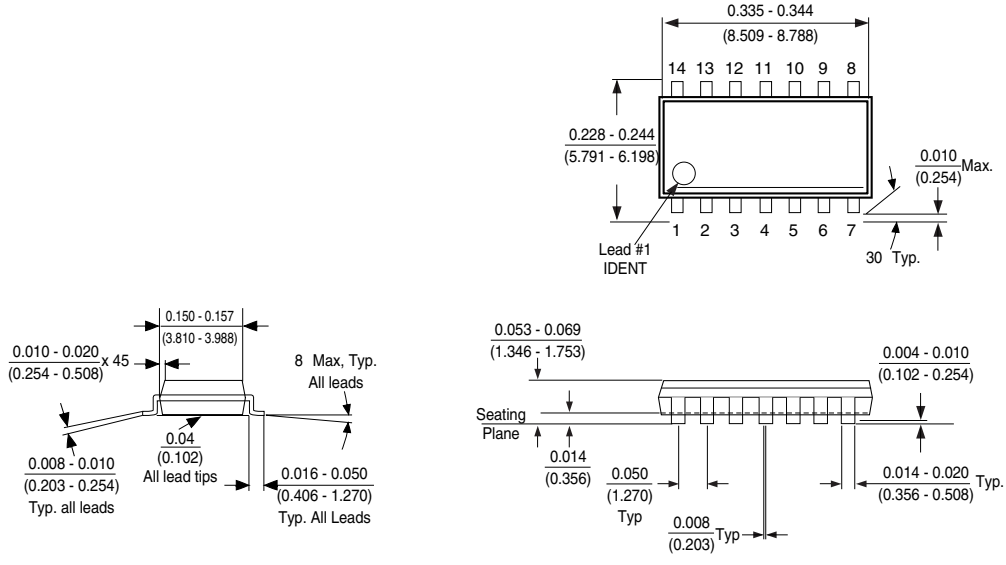
FSID	Package	Supply Voltage	Temperature Range	Packaging Option	
				Method	Qty
FMS7401LEN	PDIP8	2.7V to 3.6V	-40°C to 85°C	Rail	40
FMS7401LVN	PDIP8	2.7V to 3.6V	-40°C to 125°C	Rail	40
FMS7401LEN14	PDIP14	2.7V to 3.6V	-40°C to 85°C	Rail	25
FMS7401LVN14	PDIP14	2.7V to 3.6V	-40°C to 125°C	Rail	25
FMS7401LEM8X	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 SOIC



14-Pin SOIC



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