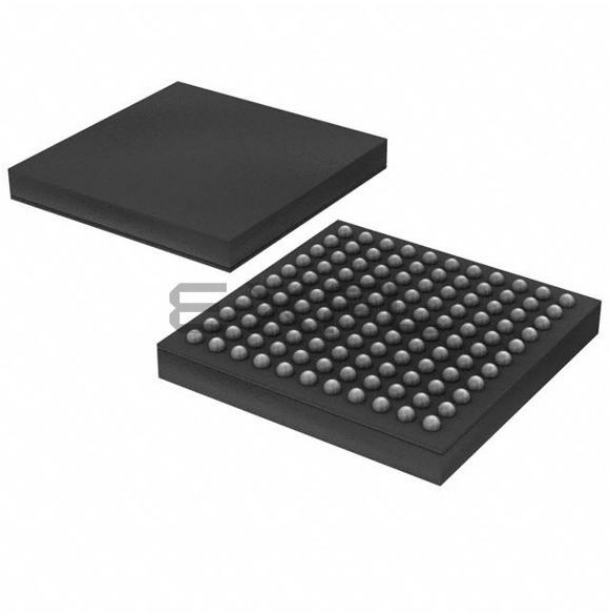


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	ARM® Cortex®-M3
Core Size	32-Bit Single-Core
Speed	48MHz
Connectivity	EBI/EMI, I²C, IrDA, SmartCard, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, LCD, POR, PWM, WDT
Number of I/O	93
Program Memory Size	512KB (512K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	128K x 8
Voltage - Supply (Vcc/Vdd)	1.98V ~ 3.8V
Data Converters	A/D 8x12b; D/A 2x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	120-VFBGA
Supplier Device Package	120-BGA (7x7)
Purchase URL	https://www.e-xfl.com/product-detail/silicon-labs/efm32gg995f512g-e-bga120r

Each Low Energy Peripheral that is clocked by LFACLK has its own prescaler setting and enable bit. The prescaler settings are configured using CMU_LFAPRESC0 and the clock enable bits can be found in CMU_LFACLKEN0. Notice that the LCD has an additional high resolution prescaler for Frame Rate Control, configured by FDIV in CMU_LCDCTRL. When operating in oversampling mode, the pulse counters are clocked by LFACLK. This is configured for each pulse counter (n) individually by setting PCNTnCLKSEL in CMU_PCNTCTRL.

11.3.1.5 LFBCLK - Low Frequency B Clock

LFBCLK is the selected clock for the Low Energy B Peripherals. There are four selectable sources for LFBCLK: LFRCO, LFXO, HFCORECLK/2 and ULFRCO. In addition, the LFBCLK can be disabled. From reset, the LFBCLK source is set to LFRCO. However, note that the LFRCO is disabled from reset. The selection is configured using the LFB field in CMU_LFCLKSEL. The HFCORECLK/2 setting allows the Low Energy B Peripherals to be used as high-frequency peripherals.

Note

If HFCORECLK/2 is selected as LFBCLK, the clock will stop in EM2/3.

Each Low Energy Peripheral that is clocked by LFBCLK has its own prescaler setting and enable bit. The prescaler settings are configured using CMU_LFBPRESC0 and the clock enable bits can be found in CMU_LFBCLKEN0.

11.3.1.6 PCNTnCLK - Pulse Counter n Clock

Each available pulse counter is driven by its own clock, PCNTnCLK where n is the pulse counter instance number. Each pulse counter can be configured to use an external pin (PCNTn_S0) or LFACLK as PCNTnCLK.

11.3.1.7 WDOGCLK - Watchdog Timer Clock

The Watchdog Timer (WDOG) can be configured to use one of three different clock sources: LFRCO, LFXO or ULFRCO. ULFRCO (Ultra Low Frequency RC Oscillator) is a separate 1 kHz RC oscillator that also runs in EM3.

11.3.1.8 AUXCLK - Auxiliary Clock

AUXCLK is a 1-28 MHz clock driven by a separate RC oscillator, AUXHFRCO. This clock is used for flash programming, and Serial Wire Output (SWO), and LESENSE operation. During flash programming, or if needed by LESENSE, this clock will be active. If the AUXHFRCO has not been enabled explicitly by software, the MSC or LESENSE module will automatically start and stop it. The AUXHFRCO is enabled by writing a 1 to AUXHFRCOEN in CMU_OSCENCMD. This explicit enabling is required when SWO is used.

11.3.2 Oscillator Selection

11.3.2.1 Start-up Time

The different oscillators have different start-up times. For the RC oscillators, the start-up time is fixed, but both the LFXO and the HFXO have configurable start-up time. At the end of the start-up time a ready flag is set to indicated that the start-up time has exceeded and that the clock is available. The low start-up time values can be used for an external clock source of already high quality, while the higher start-up times should be used when the clock signal is coming directly from a crystal. The startup time for HFXO and LFXO can be set by configuring the HFXOTIMEOUT and LFXOTIMEOUT bitfields, respectively. Both bitfields are located in CMU_CTRL. For HFXO it is also possible to enable a glitch detection filter by setting HFXOGLITCHDETEN in CMU_CTRL. The glitch detector will reset the start-up counter if a glitch is detected, making the start-up process start over again.

2. The Device responds with NAK.
3. If the application has unmasked NAK, the core generates the corresponding interrupt(s) to the application.

   The application is not required to service these interrupts, since the core takes care of rewinding of buffer pointers and re-initializing the Channel without application intervention.
4. When the Device returns an ACK, the core continues with the transfer.

Optionally, the application can utilize these interrupts. If utilized by the application:

- The NAK interrupt is masked by the application.
- The core does not generate a separate interrupt when NAK is received by the Host functionality.

Application Programming Flow

1. The application programs a channel to do a bulk transfer for a particular data size in each transaction.
   - Packet Data size can be up to 512 KBytes
   - Zero-length data must be programmed as a separate transaction.
2. Program the transfer size register with:
   - Transfer size
   - Packet Count
3. Program the DMA address.
4. Program the USB_HCx_CHAR to enable the channel.
5. The Interrupt handling by the application is as depicted in the flow diagram.

Note

The NAK interrupts are still generated internally. The application can mask off these interrupts from reaching it. The application can use these interrupts optionally.

15.4.4.1.7 Device DMA/Slave Mode Initialization

The application must meed the following conditions to set up the device core to handle traffic.

- In Slave mode, USB_GINTMSK.NPTXFEMPMSK, and USB_GINTMSK.RXFLVLMSK must be unset.
- In DMA mode, the aforementioned interrupts must be masked.

15.4.4.1.8 Transfer Stop Process

When the core is operating as a device, use the following programing sequence if you want to stop any transfers (because of an interrupt from the host, typically a reset).

15.4.4.1.8.1 Transfer Stop Programming Flow for IN Endpoints

Sequence of operations:

1. Disable the IN endpoint by programming USB_DIEP0CTL/USB_DIEPx_CTL.EPDIS = 1.
2. Wait for the USB_DIEPx_INT.EPDISBLD interrupt, which indicates that the IN endpoint is completely disabled. When the EPDISBLD interrupt is asserted, the core clears the following bits:
   - USB_DIEP0CTL/USB_DIEPx_CTL.EPDIS = 0
   - USB_DIEP0CTL/USB_DIEPx_CTL.EPENA = 0
3. Flush the TX FIFO by programming the following bits:
   - USB_GRSTCTL.TXFFLSH = 1
   - USB_GRSTCTL.TXFNUM = FIFO number specific to endpoint
4. The application can start polling till USB_GRSTCTL.TXFFLSH is cleared. When this bit is cleared, it ensures that there is no data left in the TX FIFO.

15.4.4.1.8.2 Transfer Stop Programming Flow for OUT Endpoints

Sequence of operations:

1. Enable all OUT endpoints by setting USB_DOEP0CTL/USB_DOEPx_CTL.EPENA = 1.
2. Before disabling any OUT endpoint, the application must enable Global OUT NAK mode in the core, according to the instructions in Setting the Global OUT NAK (p. 295) . This ensures that data in the RX FIFO is sent to the application successfully. Set USB_DCTL.USB_DCTL.SGOUTNAK = 1.
3. Wait for the USB_GINTSTS.GOUTNAKEFF interrupt.
4. Disable all active OUT endpoints by programming the following register bits:
   - USB_DOEP0CTL/USB_DOEPx_CTL.EPENA = 1
   - USB_DOEP0CTL/USB_DOEPx_CTL.EPDIS = 1
   - USB_DOEP0CTL/USB_DOEPx_CTL.SNAK = 1
5. Wait for the USB_DOEP0INT/USB_DOEPx_INT.EPDISBLD interrupt for each OUT endpoint programmed in the previous step. The USB_DOEP0INT/USB_DOEPx_INT.EPDISBLD interrupt indicates that the corresponding OUT endpoint is completely disabled. When the EPDISBLD interrupt is asserted, the core clears the following bits:
   - USB_DOEP0CTL/USB_DOEPx_CTL.EPENA = 0
   - USB_DOEP0CTL/USB_DOEPx_CTL.EPDIS = 0

Note

The application must not flush the Rx FIFO, as the Global OUT NAK effective interrupt earlier ensures that there is no data left in the Rx FIFO.

15.4.4.2 Device Programming Operations

Table 15.2 (p. 288) provides links to the programming sequence for different USB transaction types.

15.4.4.2.3.1 Packet Write in Slave Mode

This section describes how the application writes data packets to the endpoint FIFO in Slave mode.

1. The application can either choose polling or interrupt mode.
   - In polling mode, application monitors the status of the endpoint transmit data FIFO, by reading the USB_DIEPx_TXFSTS register, to determine, if there is enough space in the data FIFO.
   - In interrupt mode, application waits for the USB_DIEPx_INT.TXFEMP interrupt and then reads the USB_DIEPx_TXFSTS register, to determine, if there is enough space in the data FIFO.
   - To write a single non-zero length data packet, there must be space to write the entire packet is the data FIFO.
   - For writing zero length packet, application must not look for FIFO space.
2. Using one of the above mentioned methods, when the application determines that there is enough space to write a transmit packet, the application must first write into the endpoint control register, before writing the data into the data FIFO. The application, typically must do a read modify write on the USB_DIEPx_CTL, to avoid modifying the contents of the register, except for setting the Endpoint Enable bit.

The application can write multiple packets for the same endpoint, into the transmit FIFO, if space is available. For periodic IN endpoints, application must write packets only for one frame. It can write packets for the next periodic transaction, only after getting transfer complete for the previous transaction.

15.4.4.2.3.2 Setting Global Non-Periodic IN Endpoint NAK

Internal Data Flow

1. When the application sets the Global Non-periodic IN NAK bit (USB_DCTL.SGNPINNAK), the core stops transmitting data on the non-periodic endpoint, irrespective of data availability in the Non-periodic Transmit FIFO.
2. Non-isochronous IN tokens receive a NAK handshake reply
3. The core asserts the USB_GINTSTS.GINNAKEFF interrupt in response to the USB_DCTL.SGNPINNAK bit.
4. Once the application detects this interrupt, it can assume that the core is in the Global Non-periodic IN NAK mode. The application can clear this interrupt by clearing the USB_DCTL.SGNPINNAK bit.

Application Programming Sequence

1. To stop transmitting any data on non-periodic IN endpoints, the application must set the USB_DCTL.SGNPINNAK bit. To set this bit, the following field must be programmed
   - USB_DCTL.SGNPINNAK = 1
2. Wait for the assertion of the USB_GINTSTS.GINNAKEFF interrupt. This interrupt indicates the core has stopped transmitting data on the non-periodic endpoints.
3. The core can transmit valid non-periodic IN data after the application has set the USB_DCTL.SGNPINNAK bit, but before the assertion of the USB_GINTSTS.GINNAKEFF interrupt.
4. The application can optionally mask this interrupt temporarily by writing to the USB_GINTMSK.GINNAKEFFMSK bit.
   - USB_GINTMSK.GINNAKEFFMSK = 0
5. To exit Global Non-periodic IN NAK mode, the application must clear the USB_DCTL.SGNPINNAK. This also clears the USB_GINTSTS.GINNAKEFF interrupt.
   - USB_DCTL.SGNPINNAK = 1
6. If the application has masked this interrupt earlier, it must be unmasked as follows:
   - USB_GINTMSK.GINNAKEFFMSK = 1

15.4.4.2.3.3 Setting IN Endpoint NAK

Internal Data Flow

Bit	Name	Reset	Access	Description
1	VREGOSL	0	W1	**Set VREGO Sense Low Interrupt Flag**
	Write to 1 to set the VREGO Sense Low Interrupt Flag.			
0	VREGOSH	0	W1	**Set VREGO Sense High Interrupt Flag**
	Write to 1 to set the VREGO Sense High Interrupt Flag.			

15.6.5 USB_IFC - Interrupt Flag Clear Register

Offset	Bit Position																													
0x010	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															W1	W1
Name																															VREGOSL	VREGOSH

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	VREGOSL	0	W1	**Clear VREGO Sense Low Interrupt Flag**
	Write to 1 to clear the VREGO Sense Low Interrupt Flag.			
0	VREGOSH	0	W1	**Clear VREGO Sense High Interrupt Flag**
	Write to 1 to clear the VREGO Sense High Interrupt Flag.			

15.6.6 USB_IEN - Interrupt Enable Register

Offset	Bit Position																													
0x014	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															RW	RW
Name																															VREGOSL	VREGOSH

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	VREGOSL	0	RW	**VREGO Sense Low Interrupt Enable**
	Enable interrupt on VREGO Sense Low.			
0	VREGOSH	0	RW	**VREGO Sense High Interrupt Enable**
	Enable interrupt on VREGO Sense High.			

Bit	Name	Reset	Access	Description
31:14	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
13:0	HAINT	0x0000	R	**Channel Interrupt for channel 0 - 13.**
	When the interrupt bit for a channel x set, one or more of the interrupt flags in the USB_HCx_INT are set.			

15.6.33 USB_HAINTMSK - Host All Channels Interrupt Mask Register

The Host All Channel Interrupt Mask register works with the Host All Channel Interrupt register to interrupt the application when an event occurs on a channel. There is one interrupt mask bit per channel. Set bits to unmask.

Offset	Bit Position																														
0x3C418	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																			0x0000												
Access																			RW												
Name																			HAINTMSK												

Bit	Name	Reset	Access	Description
31:14	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
13:0	HAINTMSK	0x0000	RW	**Channel Interrupt Mask for channel 0 - 13**
	Set bit n to unmask channel n interrupts.			

15.6.34 USB_HPRT - Host Port Control and Status Register

This register is available only in Host mode. This register holds USB port-related information such as USB reset, enable, suspend, resume, connect status, and test mode for the port. Some bits in this register can trigger an interrupt to the application through the Host Port Interrupt bit of the Core Interrupt register (USB_GINTSTS.PRTINT). On a Port Interrupt, the application must read this register and clear the bit that caused the interrupt. For the RW1H bits, the application must write a 1 to the bit to clear the interrupt.

Offset	Bit Position																														
0x3C440	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset														0x0			0x0			0	0x0			0	0	0	0	0	0	0	0	0
Access														R			RW			RW	R			RW	RW1H	RW	RW1H	R	RW1H	RW1H	RW1H	R
Name														PRTSPD			PRTTSTCTL			PRTPWR	PRTLNSTS			PRTRST	PRTSUSP	PRTRES	PRTOVRCURRCHNG	PRTOVRCURRACT	PRTENCHNG	PRTENA	PRTCONNDET	PRTCONNSTS

Bit	Name	Reset	Access	Description
31:19	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
18:17	PRTSPD	0x0	R	**Port Speed**
	Indicates the speed of the device attached to this port.			

15.6.75 USB_FIFO6Dx - Device EP 6/Host Channel 6 FIFO

This register, available in both Host and Device modes, is used to read or write the FIFO space for endpoint 6 or channel 6, in a given direction. If a host channel is of type IN, the FIFO can only be read on the channel. Similarly, if a host channel is of type OUT, the FIFO can only be written on the channel.

Offset	Bit Position																															
0x43000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xXXXXXXXX																															
Access	RW																															
Name	FIFO6D																															

Bit	Name	Reset	Access	Description
31:0	FIFO6D	0xXXXXXXXX	RW	**Device EP 6/Host Channel 6 FIFO**
	FIFO 6 push/pop region. Used in slave mode.			

15.6.76 USB_FIFO7Dx - Host Channel 7 FIFO

This register, available in Host mode, is used to read or write the FIFO space for channel 7, in a given direction. If a host channel is of type IN, the FIFO can only be read on the channel. Similarly, if a host channel is of type OUT, the FIFO can only be written on the channel.

Offset	Bit Position																															
0x44000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0xXXXXXXXX																															
Access	RW																															
Name	FIFO7D																															

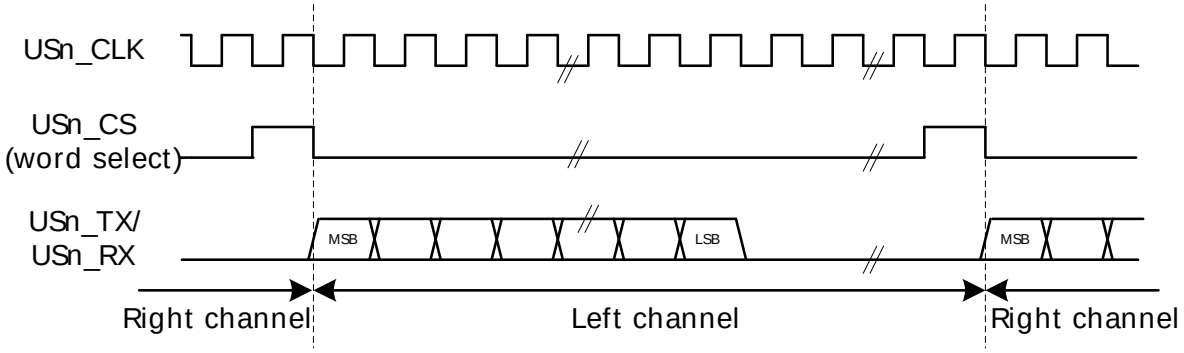
Bit	Name	Reset	Access	Description
31:0	FIFO7D	0xXXXXXXXX	RW	**Host Channel 7 FIFO**
	FIFO 7 push/pop region. Used in slave mode.			

15.6.77 USB_FIFO8Dx - Host Channel 8 FIFO

This register, available in Host mode, is used to read or write the FIFO space for channel 8, in a given direction. If a host channel is of type IN, the FIFO can only be read on the channel. Similarly, if a host channel is of type OUT, the FIFO can only be written on the channel.

In mono-mode, the word-select signal pulses at the beginning of each word instead of toggling for each word. Mono I2S waveform is shown in Figure 17.20 (p. 471) .

Figure 17.20. USART Mono I2S waveform



17.3.3.6.3 Using I2S Mode

When using the USART in I2S mode, DATABITS in USARTn_FRAME must be set to 8 or 16 data-bits. 8 databits can be used in all modes, and 16 can be used in the modes where the number of bytes in the I2S word is even. In addition to this, MSBF in USARTn_CTRL should be set, and CLKPOL and CLKPHA in USARTn_CTRL should be cleared.

The USART does not have separate TX and RX buffers for left and right data, so when using I2S in stereo mode, the application must keep track of whether the buffers contain left or right data. This can be done by observing TXBLRIGHT, RXDATAVRIGHT and RXFULLRIGHT in USARTn_STATUS. TXBLRIGHT tells whether TX is expecting data for the left or right channel. It will be set with TXBL if right data is expected. The receiver will set RXDATAVRIGHT if there is at least one right element in the buffer, and RXFULLRIGHT if the buffer is full of right elements.

When using I2S with DMA, separate DMA requests can be used for left and right data by setting DMASPLIT in USARTn_I2SCTRL.

In both master and slave mode the USART always starts transmitting on the LEFT channel after being enabled. In master mode, the transmission will stop if TX becomes empty. In that case, TXC is set. Continuing the transmission in this case will make the data-stream continue where it left off. To make the USART start on the LEFT channel after going empty, disable and re-enable TX.

17.3.4 PRS-triggered Transmissions

If a transmission must be started on an event with very little delay, the PRS system can be used to trigger the transmission. The PRS channel to use as a trigger can be selected using TSEL in USARTn_TRIGCTRL. When a positive edge is detected on this signal, the receiver is enabled if RXTEN in USARTn_TRIGCTRL is set, and the transmitter is enabled if TXTEN in USARTn_TRIGCTRL is set. Only one signal input is supported by the USART.

The AUTOTX feature can also be enabled via PRS. If an external SPI device sets a pin high when there is data to be read from the device, this signal can be routed to the USART through the PRS system and be used to make the USART clock data out of the external device. If AUTOTXTEN in USARTn_TRIGCTRL is set, the USART will transmit data whenever the PRS signal selected by TSEL is high given that there is enough room in the RX buffer for the chosen frame size. Note that if there is no data in the TX buffer when using AUTOTX, the TX underflow interrupt will be set.

AUTOTXTEN can also be combined with TXTEN to make the USART transmit a command to the external device prior to clocking out data. To do this, disable TX using the TXDIS command, load the

Table 17.10. USART IrDA Pulse Widths

IRPW	Pulse width OVS=0	Pulse width OVS=1	Pulse width OVS=2	Pulse width OVS=3
00	1/16	1/8	1/6	1/4
01	2/16	2/8	2/6	N/A
10	3/16	3/8	N/A	N/A
11	4/16	N/A	N/A	N/A

By default, no filter is enabled in the IrDA demodulator. A filter can be enabled by setting IRFILT in USARTn_IRCTRL. When the filter is enabled, an incoming pulse has to last for 4 consecutive clock cycles to be detected by the IrDA demodulator.

Note that by default, the idle value of the USART data signal is high. This means that the IrDA modulator generates negative pulses, and the IrDA demodulator expects negative pulses. To make the IrDA module use RZI signalling, both TXINV and RXINV in USARTn_CTRL must be set.

The IrDA module can also modulate a signal from the PRS system, and transmit a modulated signal to the PRS system. To use a PRS channel as transmitter source instead of the USART, set IRPRSEN in USARTn_IRCTRL high. The channel is selected by configuring IRPRSSEL in USARTn_IRCTRL.

17.4 Register Map

The offset register address is relative to the registers base address.

Offset	Name	Type	Description
0x000	USARTn_CTRL	RW	Control Register
0x004	USARTn_FRAME	RW	USART Frame Format Register
0x008	USARTn_TRIGCTRL	RW	USART Trigger Control register
0x00C	USARTn_CMD	W1	Command Register
0x010	USARTn_STATUS	R	USART Status Register
0x014	USARTn_CLKDIV	RW	Clock Control Register
0x018	USARTn_RXDATAX	R	RX Buffer Data Extended Register
0x01C	USARTn_RXDATA	R	RX Buffer Data Register
0x020	USARTn_RXDOUBLEX	R	RX Buffer Double Data Extended Register
0x024	USARTn_RXDOUBLE	R	RX FIFO Double Data Register
0x028	USARTn_RXDATAXP	R	RX Buffer Data Extended Peek Register
0x02C	USARTn_RXDOUBLEXP	R	RX Buffer Double Data Extended Peek Register
0x030	USARTn_TXDATAX	W	TX Buffer Data Extended Register
0x034	USARTn_TXDATA	W	TX Buffer Data Register
0x038	USARTn_TXDOUBLEX	W	TX Buffer Double Data Extended Register
0x03C	USARTn_TXDOUBLE	W	TX Buffer Double Data Register
0x040	USARTn_IF	R	Interrupt Flag Register
0x044	USARTn_IFS	W1	Interrupt Flag Set Register
0x048	USARTn_IFC	W1	Interrupt Flag Clear Register
0x04C	USARTn_IEN	RW	Interrupt Enable Register
0x050	USARTn_IRCTRL	RW	IrDA Control Register
0x054	USARTn_ROUTE	RW	I/O Routing Register
0x058	USARTn_INPUT	RW	USART Input Register
0x05C	USARTn_I2SCTRL	RW	I2S Control Register

17.5 Register Description

17.5.1 USARTn_CTRL - Control Register

Offset	Bit Position																														
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset		0	0	0	0x0			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0x0		0	0	0	0	0
Access		RW	RW	RW	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW		RW		RW	RW	RW	RW	RW
Name		MVDIS	AUTOTX	BYTESWAP	TXDELAY			ERRSTX	ERRSRX	ERRSDMA	BIT8DV	SKIPPERRF	SCRETRANS	SCMODE	AUTOTRI	AUTOCS	CSINV	TXINV	RXINV	TXBIL	CSMA	MSBF	CLKPHA	CLKPOL		OVS		MPAB	MPM	CCEN	LOOPBK	SYNC

Bit	Name	Reset	Access	Description
31	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
30	MVDIS	0	RW	**Majority Vote Disable**
	Disable majority vote for 16x, 8x and 6x oversampling modes.			
29	AUTOTX	0	RW	**Always Transmit When RX Not Full**

20.4 Register Map

The offset register address is relative to the registers base address.

Offset	Name	Type	Description
0x000	TIMERn_CTRL	RW	Control Register
0x004	TIMERn_CMD	W1	Command Register
0x008	TIMERn_STATUS	R	Status Register
0x00C	TIMERn_IEN	RW	Interrupt Enable Register
0x010	TIMERn_IF	R	Interrupt Flag Register
0x014	TIMERn_IFS	W1	Interrupt Flag Set Register
0x018	TIMERn_IFC	W1	Interrupt Flag Clear Register
0x01C	TIMERn_TOP	RWH	Counter Top Value Register
0x020	TIMERn_TOPB	RW	Counter Top Value Buffer Register
0x024	TIMERn_CNT	RWH	Counter Value Register
0x028	TIMERn_ROUTE	RW	I/O Routing Register
0x030	TIMERn_CC0_CTRL	RW	CC Channel Control Register
0x034	TIMERn_CC0_CCV	RWH	CC Channel Value Register
0x038	TIMERn_CC0_CCVP	R	CC Channel Value Peek Register
0x03C	TIMERn_CC0_CCVB	RWH	CC Channel Buffer Register
0x040	TIMERn_CC1_CTRL	RW	CC Channel Control Register
0x044	TIMERn_CC1_CCV	RWH	CC Channel Value Register
0x048	TIMERn_CC1_CCVP	R	CC Channel Value Peek Register
0x04C	TIMERn_CC1_CCVB	RWH	CC Channel Buffer Register
0x050	TIMERn_CC2_CTRL	RW	CC Channel Control Register
0x054	TIMERn_CC2_CCV	RWH	CC Channel Value Register
0x058	TIMERn_CC2_CCVP	R	CC Channel Value Peek Register
0x05C	TIMERn_CC2_CCVB	RWH	CC Channel Buffer Register
0x070	TIMERn_DTCTRL	RW	DTI Control Register
0x074	TIMERn_DTTIME	RW	DTI Time Control Register
0x078	TIMERn_DTFC	RW	DTI Fault Configuration Register
0x07C	TIMERn_DTOGEN	RW	DTI Output Generation Enable Register
0x080	TIMERn_DTFAULT	R	DTI Fault Register
0x084	TIMERn_DTFAULTC	W1	DTI Fault Clear Register
0x088	TIMERn_DTLOCK	RW	DTI Configuration Lock Register

Table 21.1. RTC Resolution Vs Overflow

RTC_PRESC	Resolution	Overflow
0	30,5 µs	512 s
1	61,0 µs	1024 s
2	122 µs	2048 s
3	244 µs	1,14 hours
4	488 µs	2,28 hours
5	977 µs	4,55 hours
6	1,95 ms	9,10 hours
7	3,91 ms	18,2 hours
8	7,81 ms	1,52 days
9	15,6 ms	3,03 days
10	31,25 ms	6,07 days
11	62,5 ms	12,1 days
12	0,125 s	24,3 days
13	0,25 s	48,5 days
14	0,5 s	97,1 days
15	1 s	194 days

21.3.2 Compare Channels

Two compare channels are available in the RTC. The compare values can be set by writing to the RTC compare channel registers RTC_COMPn, and when RTC_CNT is equal to one of these, the respective compare interrupt flag COMPn is set.

If COMP0TOP is set, the compare value set for compare channel 0 is used as a top value for the RTC, and the timer is cleared on a compare match with compare channel 0. If using the COMP0TOP setting, make sure to set this bit prior to or at the same time the EN bit is set. Setting COMP0TOP after the EN bit is set may cause unintended operation (i.e. if CNT > COMP0).

21.3.2.1 LETIMER Triggers

A compare event on either of the compare channels can start the LETIMER. See the LETIMER documentation for more information on this feature.

21.3.2.2 PRS Sources

Both the compare channels of the RTC can be used as PRS sources. They will generate a pulse lasting one RTC clock cycle on a compare match.

21.3.3 Interrupts

The interrupts generated by the RTC are combined into one interrupt vector. If interrupts for the RTC is enabled, an interrupt will be made if one or more of the interrupt flags in RTC_IF and their corresponding bits in RTC_IEN are set. Interrupt events are overflow and compare match on either compare channels. Clearing of an interrupt flag is performed by writing to the corresponding bit in the RTC_IFC register.

Bit	Name	Reset	Access	Description
31:24	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
23	ALTEX7PEN	0	RW	**ALTEX7 Pin Enable**
22	ALTEX6PEN	0	RW	**ALTEX6 Pin Enable**
21	ALTEX5PEN	0	RW	**ALTEX5 Pin Enable**
20	ALTEX4PEN	0	RW	**ALTEX4 Pin Enable**
19	ALTEX3PEN	0	RW	**ALTEX3 Pin Enable**
18	ALTEX2PEN	0	RW	**ALTEX2 Pin Enable**
17	ALTEX1PEN	0	RW	**ALTEX1 Pin Enable**
16	ALTEX0PEN	0	RW	**ALTEX0 Pin Enable**
15	CH15PEN	0	RW	**CH15 Pin Enable**
14	CH14PEN	0	RW	**CH14 Pin Enable**
13	CH13PEN	0	RW	**CH13 Pin Enable**
12	CH12PEN	0	RW	**CH12 Pin Enable**
11	CH11PEN	0	RW	**CH11 Pin Enable**
10	CH10PEN	0	RW	**CH10 Pin Enable**
9	CH9PEN	0	RW	**CH9 Pin Enable**
8	CH8PEN	0	RW	**CH8 Pin Enable**
7	CH7PEN	0	RW	**CH7 Pin Enable**
6	CH6PEN	0	RW	**CH6 Pin Enable**
5	CH5PEN	0	RW	**CH5 Pin Enable**
4	CH4PEN	0	RW	**CH4 Pin Enable**
3	CH3PEN	0	RW	**CH3 Pin Enable**
2	CH2PEN	0	RW	**CH2 Pin Enable**

27.4 Register Map

The offset register address is relative to the registers base address.

Offset	Name	Type	Description
0x000	VCMP_CTRL	RW	Control Register
0x004	VCMP_INPUTSEL	RW	Input Selection Register
0x008	VCMP_STATUS	R	Status Register
0x00C	VCMP_IEN	RW	Interrupt Enable Register
0x010	VCMP_IF	R	Interrupt Flag Register
0x014	VCMP_IFS	W1	Interrupt Flag Set Register
0x018	VCMP_IFC	W1	Interrupt Flag Clear Register

27.5 Register Description

27.5.1 VCMP_CTRL - Control Register

Offset	Bit Position																													
0x000	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset		1			0x7										0	0							0x0					0		0		0
Access		RW			RW										RW	RW							RW					RW		RW		RW
Name		HALFBIAS			BIASPROG										IFALL	IRISE							WARMTIME					HYSTEN		INACTVAL		EN

Bit	Name	Reset	Access	Description
31	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
30	HALFBIAS	1	RW	**Half Bias Current**
	Set this bit to 1 to halve the bias current. Table 27.1 (p. 681) .			
29:28	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
27:24	BIASPROG	0x7	RW	**VCMP Bias Programming Value**
	These bits control the bias current level. Table 27.1 (p. 681) .			
23:18	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
17	IFALL	0	RW	**Falling Edge Interrupt Sense**
	Set this bit to 1 to set the EDGE interrupt flag on falling edges of comparator output.			
16	IRISE	0	RW	**Rising Edge Interrupt Sense**
	Set this bit to 1 to set the EDGE interrupt flag on rising edges of comparator output.			
15:11	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
10:8	WARMTIME	0x0	RW	**Warm-Up Time**
	Set warm-up time			

Value	Mode	Description
0	4CYCLES	4 HFPERCLK cycles
1	8CYCLES	8 HFPERCLK cycles
2	16CYCLES	16 HFPERCLK cycles
3	32CYCLES	32 HFPERCLK cycles
4	64CYCLES	64 HFPERCLK cycles
5	128CYCLES	128 HFPERCLK cycles
6	256CYCLES	256 HFPERCLK cycles

29.5.2 DACn_STATUS - Status Register

Offset	Bit Position																													
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															R	R
Name																															CH1DV	CH0DV

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	CH1DV	0	R	**Channel 1 Data Valid**
	This bit is set high when CH1DATA is written and is set low when CH1DATA is used in conversion.			
0	CH0DV	0	R	**Channel 0 Data Valid**
	This bit is set high when CH0DATA is written and is set low when CH0DATA is used in conversion.			

29.5.3 DACn_CH0CTRL - Channel 0 Control Register

Offset	Bit Position																													
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																									0x0					0	0	0
Access																									RW					RW	RW	RW
Name																									PRSSEL					PRSEN	REFREN	EN

Bit	Name	Reset	Access	Description
31:8	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
7:4	PRSSEL	0x0	RW	**Channel 0 PRS Trigger Select**
	Select Channel 0 PRS input channel.			

Value	Mode	Description
0	PRSCH0	PRS ch 0 triggers channel 0 conversion.
1	PRSCH1	PRS ch 1 triggers channel 0 conversion.
2	PRSCH2	PRS ch 2 triggers channel 0 conversion.
3	PRSCH3	PRS ch 3 triggers channel 0 conversion.
4	PRSCH4	PRS ch 4 triggers channel 0 conversion.
5	PRSCH5	PRS ch 5 triggers channel 0 conversion.
6	PRSCH6	PRS ch 6 triggers channel 0 conversion.
7	PRSCH7	PRS ch 7 triggers channel 0 conversion.
8	PRSCH8	PRS ch 8 triggers channel 0 conversion.
9	PRSCH9	PRS ch 9 triggers channel 0 conversion.
10	PRSCH10	PRS ch 10 triggers channel 0 conversion.
11	PRSCH11	PRS ch 11 triggers channel 0 conversion.

3	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
2	PRSEN	0	RW	**Channel 0 PRS Trigger Enable**
	Select Channel 0 conversion trigger.			

Value	Description
0	Channel 0 is triggered by CH0DATA or COMBDATA write

Figure 30.9. Two Op-amp Differential Amplifier Overview

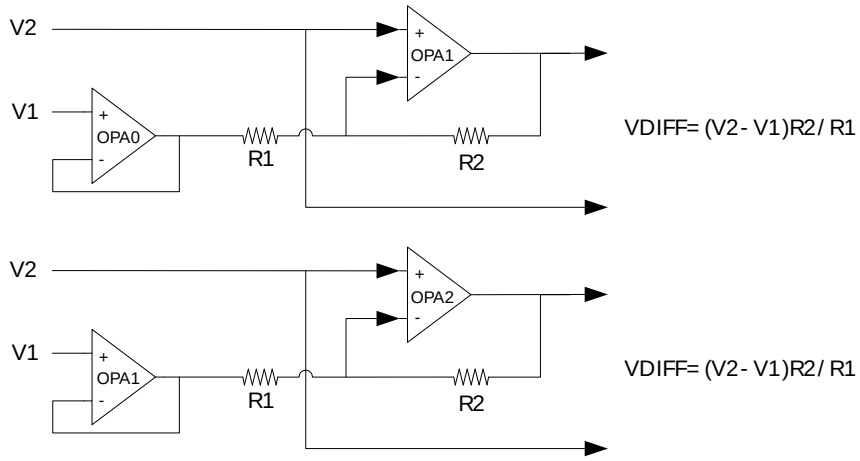


Table 30.7. OPA0/OPA1 Differential Amplifier Configuration

OPA	OPA bit-fields	OPA Configuration
OPA0	POSSEL	POSPAD1
OPA0	NEGSEL	UG
OPA0	RESINMUX	DISABLE
OPA0	NEXTOUT	1
OPA1	POSSEL	POSPAD1
OPA1	NEGSEL	OPATAP
OPA1	RESINMUX	OPA1INP

Table 30.8. OPA1/OPA2 Differential Amplifier Configuration

OPA	OPA bit-fields	OPA Configuration
OPA1	POSSEL	POSPAD1
OPA1	NEGSEL	UG
OPA1	RESINMUX	DISABLE
OPA1	NEXTOUT	1
OPA2	POSSEL	POSPAD1
OPA2	NEGSEL	OPATAP
OPA2	RESINMUX	OPA1INP

30.3.2.8 Three Opamp Differential Amplifier

This mode enables the three opamps to be internally configured to form a three opamp differential amplifier as shown in Figure 30.10 (p. 742) . Both OPA0 and OPA1 can be configured in the same unity gain mode. For both OPA0/OPA1 the positive input can be connected to any input by configuring the OPA0POSSEL/OPA1POSSEL bit-field. The OPA0/OPA1 feedback path must be configured to unity gain by setting the OPA0NEGSEL/OPA1NEGSEL bit-field to UG. In addition the OPA0RESINMUX/ OPA1RESINMUX bit-fields must be set to DISABLED. The OPA1 output must be connected to

Bit	Name	Reset	Access	Description
	Enable/disable key buffer in AES-128 mode.			
1	AES256	0	RW	**AES-256 Mode**
	Select AES-128 or AES-256 mode.			

Value	Description
0	AES-128 mode
1	AES-256 mode

Bit	Name	Reset	Access	Description
0	DECRYPT	0	RW	**Decryption/Encryption Mode**
	Select encryption or decryption.			

Value	Description
0	AES Encryption
1	AES Decryption

31.5.2 AES_CMD - Command Register

Offset	Bit Position																														
0x004	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															0	0
Access																															W1	W1
Name																															STOP	START

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	STOP	0	W1	**Encryption/Decryption Stop**
	Set to stop encryption/decryption.			
0	START	0	W1	**Encryption/Decryption Start**
	Set to start encryption/decryption.			

31.5.3 AES_STATUS - Status Register

Offset	Bit Position																														
0x008	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																																0
Access																																R
Name																																RUNNING

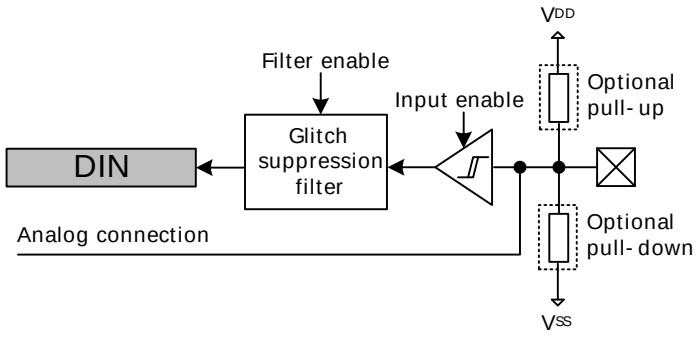
Bit	Name	Reset	Access	Description
31:1	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
0	RUNNING	0	R	**AES Running**
	This bit indicates that the AES module is running an encryption/decryption.			

MODEn	Input	Output	DOUT	Pull-down	Pull-up	Alt. strength	Input Filter	Description
0b0011			0	On			On	Input enabled with pull-down and filter
			1		On		On	Input enabled with pull-up and filter
0b0100		Push-pull	x					Push-pull
0b0101			x			On		Push-pull with alt. drive strength
0b0110		Open Source (Wired-OR)	x					Open-source
0b0111			x	On				Open-source with pull-down
0b1000		Open Drain (Wired-AND)	x					Open-drain
0b1001			x				On	Open-drain with filter
0b1010			x		On			Open-drain with pull-up
0b1011			x		On		On	Open-drain with pull-up and filter
0b1100			x			On		Open-drain with alt. drive strength
0b1101			x			On	On	Open-drain with alt. drive strength and filter
0b1110			x		On	On		Open-drain with alt. drive strength and pull-up
0b1111			x		On	On	On	Open-drain with alt. drive strength, pull-up and filter

MODEn determines which mode the pin is in at a given time. Setting MODEn to 0b0000 disables the pin, reducing power consumption to a minimum. When the output driver is disabled, the pin can be used as a connection for an analog module (e.g. ADC). Input is enabled by setting MODEn to any value other than 0b0000. The pull-up, pull-down and filter function can optionally be applied to the input, see Figure 32.2 (p. 759) .

The internal pull-up resistance, R_{PU}, and pull-down resistance, R_{PD}, are defined in the device datasheet. When the filter is enabled it suppresses glitches with pulse widths as defined by the parameter $t_{IOGLITCH}$ in the device datasheet.

Figure 32.2. Tristated Output with Optional Pull-up or Pull-down



When MODEn=0b0100 or MODEn=0b0101, the pin operates in push-pull mode. In this mode, the pin is driven either high or low, dependent on the value of GPIO_Px_DOUT. The push-pull configuration is shown in Figure 32.3 (p. 760) .

Bit	Name	Reset	Access	Description
	Enable Serial Wire Clock connection to pin. WARNING: When this pin is disabled, the device can no longer be accessed by a debugger. A reset will set the pin back to a default state as enabled. If you disable this pin, make sure you have at least a 3 second timeout at the start of you program code before you disable the pin. This way, the debugger will have time to halt the device after a reset before the pin is disabled.			

32.5.19 GPIO_INSENSE - Input Sense Register

Offset	Bit Position																													
0x124	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																															1	1
Access																															RW	RW
Name																															PRS	INT

Bit	Name	Reset	Access	Description
31:2	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
1	PRS	1	RW	**PRS Sense Enable**
	Set this bit to enable input sensing for PRS.			
0	INT	1	RW	**Interrupt Sense Enable**
	Set this bit to enable input sensing for interrupts.			

32.5.20 GPIO_LOCK - Configuration Lock Register

Offset	Bit Position																													
0x128	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset																	0x0000															
Access																	RW															
Name																	LOCKKEY															

Bit	Name	Reset	Access	Description
31:16	*Reserved*	*To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)*		
15:0	LOCKKEY	0x0000	RW	**Configuration Lock Key**
	Write any other value than the unlock code to lock MODEL, MODEH, CTRL, PINLOCKN, EPISELL, EIPSELH, INSENSE and SWDPROUTE from editing. Write the unlock code to unlock. When reading the register, bit 0 is set when the lock is enabled.			

Mode	Value	Description
Read Operation		
UNLOCKED	0	GPIO registers are unlocked
LOCKED	1	GPIO registers are locked
Write Operation		
LOCK	0	Lock GPIO registers
UNLOCK	0xA534	Unlock GPIO registers

34.3 Revision 1.00

August 28th, 2013

Updated 1 MHz band bitfield description in CMU_HFRCOBAND and CMU_AUXBAND register.

Updated 7 MHz band bitfield description in CMU_HFRCOBAND and CMU_AUXBAND register.

Updated 1 MHz HFRCOBAND and AUXHFRCOBAND to 1.2 MHz in the DI table.

Updated 7 MHz HFRCOBAND and AUXHFRCOBAND to 6.6 MHz in the DI table.

Updated Opamp System Overview description and block diagram.

Updated LETIMER Async Support in Reflex Producers table.

Updated the I2C Clock Mode table and added the Maximum Data Hold Time formula.

Added the minimum HFPERCLK requirement for I2C Slave Operation.

Added a new register access type RW1H.

Updated RMU Reset Cause Register Interpretation table.

Updated the register description of CMU_CTRL.

Updated CMU_CALCNT description.

Updated DMA_CHENC register description.

Updated description of number of wait-states for Immediate Synchronization.

Updated description of the Excite Phase timing in LESENSE.

Updated the LETIMER PRS description.

Updated OPAMP description.

Updated the EM4 with RTC and Data Retention with BURTC description.

Added LPFMODE recommendation for the ADC Input Filtering.

Updated the LETIMER description for usage in EM3.

Updated the RTC description for usage in EM3.

Updated WRITEONCE bitfield description in MSC_WRITECMD register.

Updated the MSC_TIMEBASE register description.

Updated the DMA and USB DMA access description.

Document changed status from "Preliminary".

Updated trademark, disclaimer and contact information.

Other minor corrections.

34.4 Revision 0.96

April 24th, 2012