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	AVR
Core Size	32-Bit Single-Core
Speed	50MHz
Connectivity	I²C, SPI, UART/USART, USB
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	51
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	64-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/atmel/atuc128l3u-aur

7.7.27 Performance Channel 1 Write Data Cycles

Name: PWDATA1

Access Type: Read-only

Offset: 0x828

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DATA[31:24]							

23	22	21	20	19	18	17	16
DATA[23:16]							

15	14	13	12	11	10	9	8
DATA[15:8]							

7	6	5	4	3	2	1	0
DATA[7:0]							

- **DATA: Data Cycles Counted Since Last Reset**
  Clock cycles are counted using the CLK_PDCA_HSB clock

9.8.5 Flash Version Register

Name: FVR

Access Type: Read-only

Offset: 0x10

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant Number**
  Reserved. No functionality associated.
- **VERSION: Version Number**
  Version number of the module. No functionality associated.

12.6 User Interface

Table 12-1. INTC Register Memory Map

Offset	Register	Register Name	Access	Reset
0x000	Interrupt Priority Register 0	IPR0	Read/Write	0x00000000
0x004	Interrupt Priority Register 1	IPR1	Read/Write	0x00000000
...	...	...	...	...
0x0FC	Interrupt Priority Register 63	IPR63	Read/Write	0x00000000
0x100	Interrupt Request Register 0	IRR0	Read-only	N/A
0x104	Interrupt Request Register 1	IRR1	Read-only	N/A
...	...	...	...	...
0x1FC	Interrupt Request Register 63	IRR63	Read-only	N/A
0x200	Interrupt Cause Register 3	ICR3	Read-only	N/A
0x204	Interrupt Cause Register 2	ICR2	Read-only	N/A
0x208	Interrupt Cause Register 1	ICR1	Read-only	N/A
0x20C	Interrupt Cause Register 0	ICR0	Read-only	N/A

12.6.2 Interrupt Request Registers

Name: IRR0...IRR63

Access Type: Read-only

Offset: 0x0FF - 0x1FC

Reset Value: N/A

31	30	29	28	27	26	25	24
IRR[32*x+31]	IRR[32*x+30]	IRR[32*x+29]	IRR[32*x+28]	IRR[32*x+27]	IRR[32*x+26]	IRR[32*x+25]	IRR[32*x+24]
23	22	21	20	19	18	17	16
IRR[32*x+23]	IRR[32*x+22]	IRR[32*x+21]	IRR[32*x+20]	IRR[32*x+19]	IRR[32*x+18]	IRR[32*x+17]	IRR[32*x+16]
15	14	13	12	11	10	9	8
IRR[32*x+15]	IRR[32*x+14]	IRR[32*x+13]	IRR[32*x+12]	IRR[32*x+11]	IRR[32*x+10]	IRR[32*x+9]	IRR[32*x+8]
7	6	5	4	3	2	1	0
IRR[32*x+7]	IRR[32*x+6]	IRR[32*x+5]	IRR[32*x+4]	IRR[32*x+3]	IRR[32*x+2]	IRR[32*x+1]	IRR[32*x+0]

- **IRR: Interrupt Request line**

  This bit is cleared when no interrupt request is pending on this input request line.

  This bit is set when an interrupt request is pending on this input request line.

  The are 64 IRRs, one for each group. Each IRR has 32 bits, one for each possible interrupt request, for a total of 2048 possible input lines. The IRRs are read by the software interrupt handler in order to determine which interrupt request is pending. The IRRs are sampled continuously, and are read-only.

13.7.1 Main Clock Control

Name: MCCTRL

Access Type: Read/Write

Offset: 0x000

Reset Value: 0x00000000

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
-	-	-	-	-	MCSEL		

- **MCSEL: Main Clock Select**

Table 13-8. Main clocks in ATUC64/128/256L3/4U.

MCSEL[2:0]	Main clock source
0	System RC oscillator (RCSYS)
1	Oscillator0 (OSC0)
2	DFLL
3	120MHz RC oscillator (RC120M)(1)

Note: 1. If the 120MHz RC oscillator is selected as main clock source, it must be divided by at least 4 before being used as clock source for the CPU. This division is selected by writing to the CPUSEL and CPUDIV bits in the CPUSEL register, before switching to RC120M as main clock source.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

14.6.12 DFLLn Maximum Step Register

Name: DFLLnSTEP

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	FSTEP[8]

23	22	21	20	19	18	17	16
FSTEP[7:0]							

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
CSTEP[7:0]							

- **FSTEP: Fine Maximum Step**
  This indicates the maximum step size during fine adjustment in closed-loop mode. When adjusting to a new frequency, the expected overshoot of that frequency depends on this step size.
- **CSTEP: Coarse Maximum Step**
  This indicates the maximum step size during coarse adjustment in closed-loop mode. When adjusting to a new frequency, the expected overshoot of that frequency depends on this step size.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

14.6.20 Temperature Sensor Configuration Register

Name: TSENS

Access Type: Read/Write

Reset Value: 0x00000000

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
-	-	-			-	-	EN

- **EN: Temperature Sensor Enable**
  0: The Temperature Sensor is disabled.
  1: The Temperature Sensor is enabled.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

14.6.24 Generic Clock Control

Name: GCCTRL

Access Type: Read/Write

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
DIV[15:8]							

23	22	21	20	19	18	17	16
DIV[7:0]							

15	14	13	12	11	10	9	8
-	-	-	OSCSEL[4:0]				

7	6	5	4	3	2	1	0
-	-	-	-	-	-	DIVEN	CEN

There is one GCCTRL register per generic clock in the design.

- **DIV: Division Factor**
  The number of DIV bits for each generic clock is as shown in the “Generic Clock number of DIV bits” table in the SCIF Module Configuration section.
- **OSCSEL: Oscillator Select**
  Selects the source clock for the generic clock. Please refer to the “Generic Clock Sources” table in the SCIF Module Configuration section.
- **DIVEN: Divide Enable**
  0: The generic clock equals the undivided source clock.
  1: The generic clock equals the source clock divided by 2*(DIV+1).
- **CEN: Clock Enable**
  0: The generic clock is disabled.
  1: The generic clock is enabled.

14.6.32 PLL Version Register

Name: PLLVERSION

Access Type: Read-only

Offset: 0x03C4

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	VARIANT			

15	14	13	12	11	10	9	8
-	-	-	-	VERSION[11:8]			

7	6	5	4	3	2	1	0
VERSION[7:0]							

- **VARIANT: Variant number**
  Reserved. No functionality associated.
- **VERSION: Version number**
  Version number of the module. No functionality associated.

15.5.2.3 *Calendar operation*

When the CAL bit in the Control Register is one, the counter operates in calendar mode. Before this mode is enabled, the prescaler should be set up to give a pulse every second. The date and time can then be read from or written to the Calendar Value (CALV) register.

Time is reported as seconds, minutes, and hours according to the 24-hour clock format. Date is the numeral date of month (starting on 1). Month is the numeral month of the year (1 = January, 2 = February, etc.). Year is a 6-bit field counting the offset from a software-defined leap year (e.g. 2000). The date is automatically compensated for leap years, assuming every year divisible by 4 is a leap year.

All peripheral events and interrupts work the same way in calendar mode as in counter mode. However, the Alarm Register (ARn) must be written in time/date format for the alarm to trigger correctly.

15.5.3 Interrupts

The AST can generate five separate interrupt requests:

- OVF: OVF
- PER: PER0, PER1
- ALARM: ALARM0, ALARM1
- CLKREADY
- READY

This allows the user to allocate separate handlers and priorities to the different interrupt types.

The generation of the PER interrupt is described in Section 15.5.3.1., and the generation of the ALARM interrupt is described in Section 15.5.3.2. The OVF interrupt is generated when the counter overflows, or when the alarm value is reached, if the Clear on Alarm bit in the Control Register is one. The CLKREADY interrupt is generated when SR.CLKBUSY has a 1-to-0 transition, and indicates that the clock synchronization is completed. The READY interrupt is generated when SR.BUSY has a 1-to-0 transition, and indicates that the synchronization described in Section 15.5.8 is completed.

An interrupt request will be generated if the corresponding bit in the Interrupt Mask Register (IMR) is set. Bits in IMR are set by writing a one to the corresponding bit in the Interrupt Enable Register (IER), and cleared by writing a one to the corresponding bit in the Interrupt Disable Register (IDR). The interrupt request remains active until the corresponding bit in SR is cleared by writing a one to the corresponding bit in the Status Clear Register (SCR).

The AST interrupts can wake the CPU from any sleep mode where the source clock and the interrupt controller is active.

15.5.3.1 *Periodic interrupt*

The AST can generate periodic interrupts. If the PERn bit in the Interrupt Mask Register (IMR) is one, the AST will generate an interrupt request on the 0-to-1 transition of the selected bit in the

15.6.14 Digital Tuner Register

Name: DTR

Access Type: Read/Write

Offset: 0x44

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
-	-	ADD	EXP				

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **VALUE:**
  0: The frequency is unchanged.
  1-255: The frequency will be adjusted according to the formula below.

- **ADD:**
  0: The resulting frequency is $f0\left(1 - \frac{1}{roundup\left(\frac{256}{VALUE}\right) \cdot 2^{(EXP)} + 1}\right)$ for $VALUE > 0$.

  1: The resulting frequency is $f0\left(1 + \frac{1}{roundup\left(\frac{256}{VALUE}\right) \cdot 2^{(EXP)} - 1}\right)$ for $VALUE > 0$.

- **EXP:**
  The frequency will be adjusted according to the formula above.

Table 19-2. GPIO Register Memory Map

Offset	Register	Function	Register Name	Access	Reset	Config. Protection	Access Protection
0x02C	Peripheral Mux Register 1	Toggle	PMR1T	Write-only		Y	N
0x030	Peripheral Mux Register 2	Read/Write	PMR2	Read/Write	-(1)	Y	N
0x034	Peripheral Mux Register 2	Set	PMR2S	Write-only		Y	N
0x038	Peripheral Mux Register 2	Clear	PMR2C	Write-only		Y	N
0x03C	Peripheral Mux Register 2	Toggle	PMR2T	Write-only		Y	N
0x040	Output Driver Enable Register	Read/Write	ODER	Read/Write	-(1)	Y	N
0x044	Output Driver Enable Register	Set	ODERS	Write-only		Y	N
0x048	Output Driver Enable Register	Clear	ODERC	Write-only		Y	N
0x04C	Output Driver Enable Register	Toggle	ODERT	Write-only		Y	N
0x050	Output Value Register	Read/Write	OVR	Read/Write	-(1)	N	N
0x054	Output Value Register	Set	OVRS	Write-only		N	N
0x058	Output Value Register	Clear	OVRC	Write-only		N	N
0x05c	Output Value Register	Toggle	OVRT	Write-only		N	N
0x060	Pin Value Register	Read	PVR	Read-only	Depending on pin states	N	N
0x064	Pin Value Register	-	-	-		N	N
0x068	Pin Value Register	-	-	-		N	N
0x06c	Pin Value Register	-	-	-		N	N
0x070	Pull-up Enable Register	Read/Write	PUER	Read/Write	-(1)	Y	N
0x074	Pull-up Enable Register	Set	PUERS	Write-only		Y	N
0x078	Pull-up Enable Register	Clear	PUERC	Write-only		Y	N
0x07C	Pull-up Enable Register	Toggle	PUERT	Write-only		Y	N
0x090	Interrupt Enable Register	Read/Write	IER	Read/Write	-(1)	N	N
0x094	Interrupt Enable Register	Set	IERS	Write-only		N	N
0x098	Interrupt Enable Register	Clear	IERC	Write-only		N	N
0x09C	Interrupt Enable Register	Toggle	IERT	Write-only		N	N
0x0A0	Interrupt Mode Register 0	Read/Write	IMR0	Read/Write	-(1)	N	N
0x0A4	Interrupt Mode Register 0	Set	IMR0S	Write-only		N	N
0x0A8	Interrupt Mode Register 0	Clear	IMR0C	Write-only		N	N
0x0AC	Interrupt Mode Register 0	Toggle	IMR0T	Write-only		N	N
0x0B0	Interrupt Mode Register 1	Read/Write	IMR1	Read/Write	-(1)	N	N
0x0B4	Interrupt Mode Register 1	Set	IMR1S	Write-only		N	N
0x0B8	Interrupt Mode Register 1	Clear	IMR1C	Write-only		N	N
0x0BC	Interrupt Mode Register 1	Toggle	IMR1T	Write-only		N	N

20.7.8 Transmitter Holding Register

Name: THR

Access Type: Write-only

Offset: 0x1C

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	TXCHR[8]

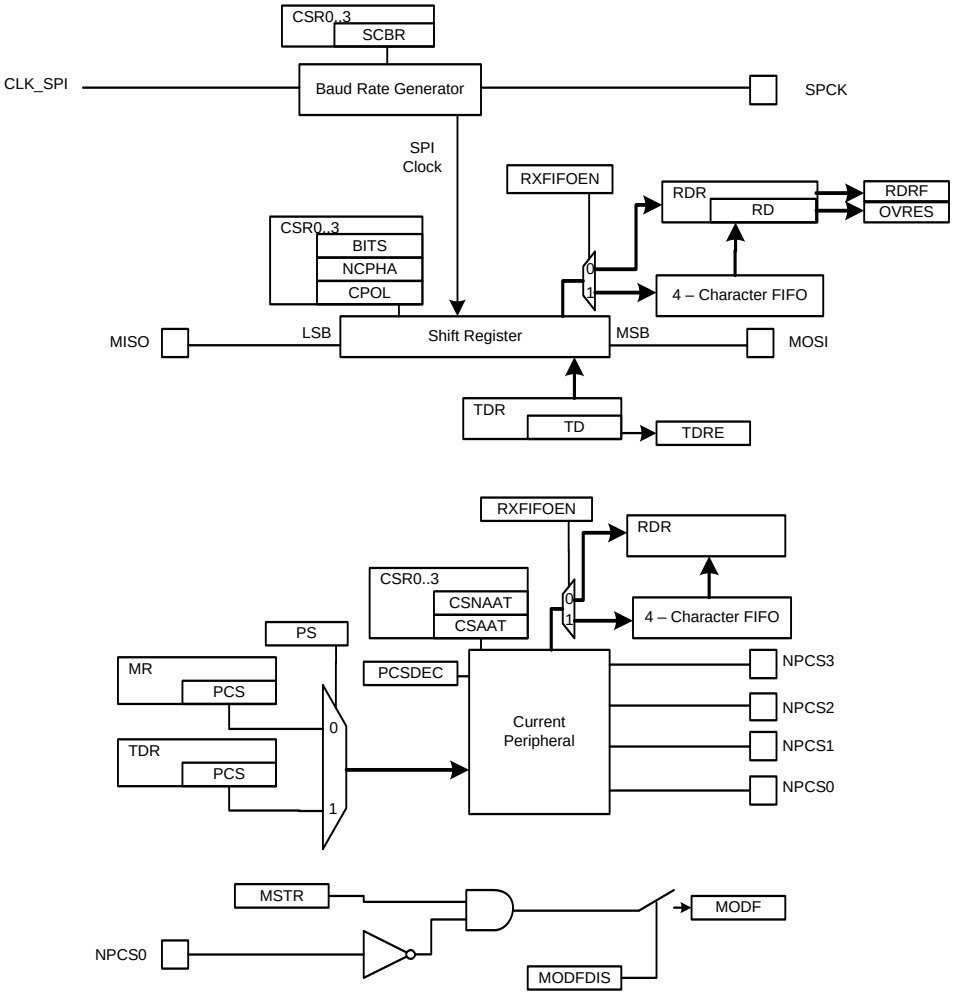
7	6	5	4	3	2	1	0
TXCHR[7:0]							

- **TXCHR: Character to be Transmitted**
  If TXRDY is zero this field contains the next character to be transmitted.

Figure 21-5 on page 491shows a block diagram of the SPI when operating in master mode. Figure 21-6 on page 492 shows a flow chart describing how transfers are handled.

21.7.3.1 Master mode block diagram

Figure 21-5. Master Mode Block Diagram



25.7.8 Status Register

Name: SR

Access Type: Read-only

Offset: 0x1C

Reset Value: 0x00000000

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
-	-	-	-	BUSY	READY	-	TOFL

- **BUSY: Interface Busy**
  This bit is automatically cleared when the interface is no longer busy.
  This bit is set when the user interface is busy and will not respond to new write operations.
- **READY: Interface Ready**
  This bit is cleared by writing a one to the corresponding bit in the SCR register.
  This bit is set when the BUSY bit has a 1-to-0 transition.
- **TOFL: Timebase Overflow**
  This bit is cleared by writing a one to corresponding bit in the SCR register.
  This bit is set when the timebase counter has wrapped at its top value.

tem will only be woken up if the user peripheral generates an interrupt as a result of the operation. This concept is known as SleepWalking and is described in further detail in the Power Manager chapter. Note that asynchronous peripheral events may be associated with a delay due to the need to restart the system clock source if this has been stopped in the sleep mode.

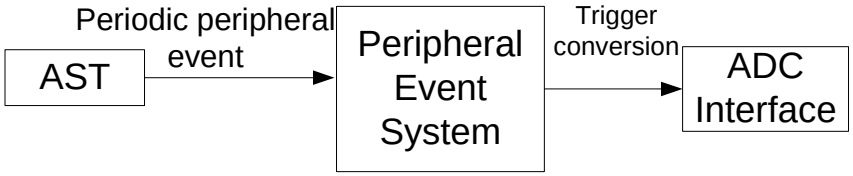
27.5 Application Example

This application example shows how the Peripheral Event System can be used to program the ADC Interface to perform ADC conversions at selected intervals.

Conversions of the active analog channels are started with a software or a hardware trigger. One of the possible hardware triggers is a peripheral event trigger, allowing the Peripheral Event System to synchronize conversion with some configured peripheral event source. From Table 27-3 and Table 27-4, it can be read that this peripheral event source can be either an AST peripheral event, or an event from the PWM Controller. The AST can generate periodic peripheral events at selected intervals, among other types of peripheral events. The Peripheral Event System can then be used to set up the ADC Interface to sample an analog signal at regular intervals.

The user must enable peripheral events in the AST and in the ADC Interface to accomplish this. The periodic peripheral event in the AST is enabled by writing a one to the corresponding bit in the AST Event Enable Register (EVE). To select the peripheral event trigger for the ADC Interface, the user must write the value 0x7 to the Trigger Mode (TRGMOD) field in the ADC Interface Trigger Register (TRGR). When the peripheral events are enabled, the AST will generate peripheral events at the selected intervals, and the Peripheral Event System will route the peripheral events to the ADC Interface, which will perform ADC conversions at the selected intervals.

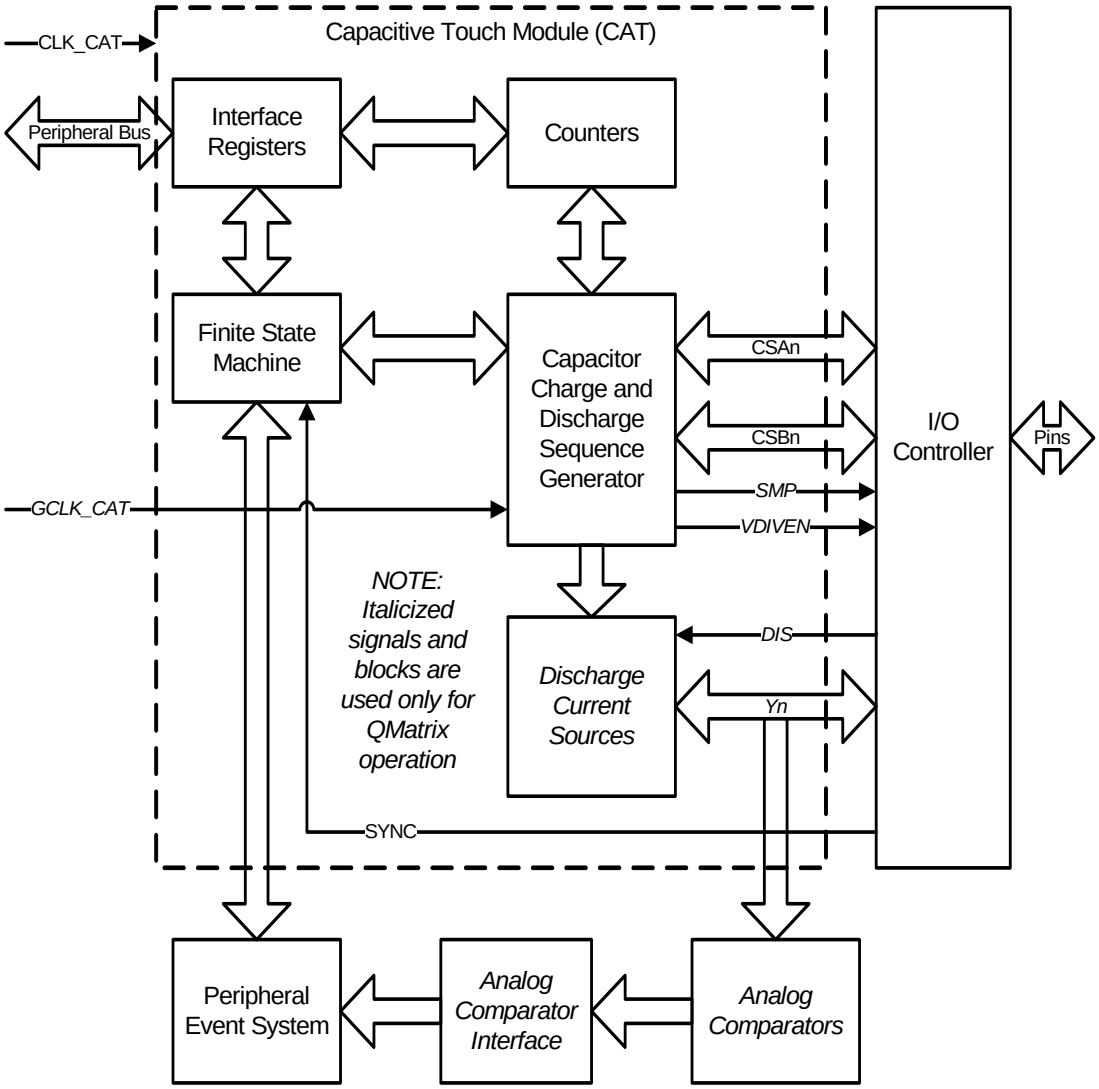
Figure 27-2. Application Example



Since the AST peripheral event is asynchronous, the description above will also work in sleep modes where the ADC clock is stopped. In this case, the ADC clock (and clock source, if needed) will be restarted during the ADC conversion. After the conversion, the ADC clock and clock source will return to the sleep state, unless the ADC generates an interrupt, which in turn will wake up the system. Using asynchronous interrupts thus allows ADC operation in much lower power states than would otherwise be possible.

31.3 Block Diagram

Figure 31-1. CAT Block Diagram



31.4 I/O Lines Description

Table 31-1. I/O Lines Description

Name	Description	Type
CSAn	Capacitive sense A line n	I/O
CSBn	Capacitive sense B line n	I/O
DIS	Discharge current control (only used for QMatrix)	Analog

34. Programming and Debugging

34.1 Overview

The ATUC64/128/256L3/4U supports programming and debugging through two interfaces, JTAG or aWire. JTAG is an industry standard interface and allows boundary scan for PCB testing, as well as daisy-chaining of multiple devices on the PCB. aWire is an Atmel proprietary protocol which offers higher throughput and robust communication, and does not require application pins to be reserved. Either interface provides access to the internal Service Access Bus (SAB), which offers a bridge to the High Speed Bus, giving access to memories and peripherals in the device. By using this bridge to the bus system, the flash and fuses can thus be programmed by accessing the Flash Controller in the same manner as the CPU.

The SAB also provides access to the Nexus-compliant On-chip Debug (OCD) system in the device, which gives the user non-intrusive run-time control of the program execution. Additionally, trace information can be output on the Auxiliary (AUX) debug port or buffered in internal RAM for later retrieval by JTAG or aWire.

34.2 Service Access Bus

The AVR32 architecture offers a common interface for access to On-chip Debug, programming, and test functions. These are mapped on a common bus called the Service Access Bus (SAB), which is linked to the JTAG and aWire port through a bus master module, which also handles synchronization between the debugger and SAB clocks.

When accessing the SAB through the debugger there are no limitations on debugger frequency compared to chip frequency, although there must be an active system clock in order for the SAB accesses to complete. If the system clock is switched off in sleep mode, activity on the debugger will restart the system clock automatically, without waking the device from sleep. Debuggers may optimize the transfer rate by adjusting the frequency in relation to the system clock. This ratio can be measured with debug protocol specific instructions.

The Service Access Bus uses 36 address bits to address memory or registers in any of the slaves on the bus. The bus supports sized accesses of bytes (8 bits), halfwords (16 bits), or words (32 bits). All accesses must be aligned to the size of the access, i.e. halfword accesses must have the lowest address bit cleared, and word accesses must have the two lowest address bits cleared.

34.2.1 SAB Address Map

The SAB gives the user access to the internal address space and other features through a 36 bits address space. The 4 MSBs identify the slave number, while the 32 LSBs are decoded within the slave's address space. The SAB slaves are shown in Table 34-1.

Table 34-1. SAB Slaves, Addresses and Descriptions

Slave	Address [35:32]	Description
Unallocated	0x0	Intentionally unallocated
OCD	0x1	OCD registers
HSB	0x4	HSB memory space, as seen by the CPU

35.6.6 32kHz RC Oscillator (RC32K) Characteristics

Table 35-16. 32kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency[1]		20	32	44	kHz
I_{RC32K}	Current consumption			0.7		µA
$t_{STARTUP}$	Startup time[1]			100		µs

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

35.6.7 System RC Oscillator (RCSYS) Characteristics

Table 35-17. System RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OUT}	Output frequency	Calibrated at 85°C	111.6	115	118.4	kHz

35.7 Flash Characteristics

Table 35-18 gives the device maximum operating frequency depending on the number of flash wait states and the flash read mode. The FSW bit in the FLASHCDW FSR register controls the number of wait states used when accessing the flash memory.

Table 35-18. Maximum Operating Frequency

Flash Wait States	Read Mode	Maximum Operating Frequency
1	High speed read mode	50MHz
0		25MHz
1	Normal read mode	30MHz
0		15MHz

Table 35-19. Flash Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
t_{FPP}	Page programming time	f_{CLK_HSB} = 50MHz		5		ms
t_{FPE}	Page erase time			5		
t_{FFP}	Fuse programming time			1		
t_{FEA}	Full chip erase time (EA)			6		
t_{FCE}	JTAG chip erase time (CHIP_ERASE)	f_{CLK_HSB} = 115kHz		310		

35.10 Timing Characteristics

35.10.1 Startup, Reset, and Wake-up Timing

The startup, reset, and wake-up timings are calculated using the following formula:

$$t = t_{CONST} + N_{CPU} \times t_{CPU}$$

Where t_{CONST} and N_{CPU} are found in Table 35-39. t_{CPU} is the period of the CPU clock. If a clock source other than RCSYS is selected as the CPU clock, the oscillator startup time, $t_{OSCSTART}$, must be added to the wake-up time from the stop, deepstop, and static sleep modes. Please refer to the source for the CPU clock in the "Oscillator Characteristics" on page 905 for more details about oscillator startup times.

Table 35-39. Maximum Reset and Wake-up Timing[1]

Parameter		Measuring	Max t_{CONST} (in µs)	Max N_{CPU}
Startup time from power-up, using regulator		Time from VDDIN crossing the V_{POT+} threshold of POR33 to the first instruction entering the decode stage of CPU. VDDCORE is supplied by the internal regulator.	2210	0
Startup time from power-up, no regulator		Time from VDDIN crossing the V_{POT+} threshold of POR33 to the first instruction entering the decode stage of CPU. VDDCORE is connected to VDDIN.	1810	0
Startup time from reset release		Time from releasing a reset source (except POR18, POR33, and SM33) to the first instruction entering the decode stage of CPU.	170	0
Wake-up	Idle	From wake-up event to the first instruction of an interrupt routine entering the decode stage of the CPU.	0	19
	Frozen		0	110
	Standby		0	110
	Stop		27 + $t_{OSCSTART}$	116
	Deepstop		27 + $t_{OSCSTART}$	116
	Static		97 + $t_{OSCSTART}$	116
Wake-up from shutdown		From wake-up event to the first instruction entering the decode stage of the CPU.	1180	0

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.

35.10.2 RESET_N Timing

Table 35-40. RESET_N Waveform Parameters[1]

Symbol	Parameter	Conditions	Min	Max	Units
t_{RESET}	RESET_N minimum pulse length		10		ns

Note: 1. These values are based on simulation and characterization of other AVR microcontrollers manufactured in the same process technology. These values are not covered by test limits in production.