ECEN 5823-001
Internet of Things Embedded Firmware

Lecture #3

04 September 2018

Agenda

- Class Announcements
- TA office hours
- Reading list
- Course Survey results
- Quiz #1 review
- Low Energy versus Low Power
- Clocking
- Managing Energy Modes
- Interrupts

Class Announcements

- The class currently has x students on the waiting list
- Quiz #2 is due at 11:59 on Sunday, September 8th, 2018
- Simplicity Exercise due, Wednesday, September 5th, at 11:59pm via Canvas
- If you need a Blue Gecko development kit, please arrange a time to pick one up at my office
- Simplicity IDE demo hosted by Vipul or Gunj
  - Wednesday the 5th and Thursday the 6th
  - ESE Lab, 1B24
  - 6:00 to 7:30pm

Simplicity Exercise update

1. How much current does a single LED draw when the output drive is set to “Strong” with the original code?

2. After commenting out the standard output drive and uncommenting “Weak” drive, how much current does a single LED draw?

3. Is there a difference in current between the answers for question 1 and 2? And, explain your answer, why or why not?
   a. Due to measurement accuracy, a difference is defined as currents measured with a difference greater than 75uA

4. Using the Energy Profiler with “weak” drive LEDs, what is ~~the Energy Score and~~ average current measured before commenting out turning on LED1?

5. Using the Energy Profiler with “weak” drive LEDs, what is ~~the Energy Score and~~ average current measured after commenting out turning on LED1?

TA Office Hours

- Gunj:
  - Monday: 4-6pm
  - Thursday: 4-6pm
- Vipul:
  - Wednesdays 12.30pm - 2.30pm
  - Fridays 4:00pm - 6:00pm
- Where?
  - ESE lab

Reading List

Questions from these readings plus the lectures from August 28th, 2018 onward will be on the weekly quiz.

“Testing and Debugging Concurrency Bugs in Event-Driven Programs,” Guy Martin Tchamgoue, Kyong-Hoon Kim, and Yong-Kee Jim http://www.sersc.org/journals/IJAST/vol40/4.pdf
Bluetooth 4.1, 4.2 and 5 Compatible Bluetooth Low Energy SoCs and Tools Meet IoT Challenges (Part1)
https://www.digikey.com/en/articles/techzone/2017/apr/bluetooth-41-42-5-low-energy-socs-meet-iot-challenges-part-1

Recommended readings. These readings will not be on the weekly quiz, but will be helpful in the class programming assignments.

“Silicon Labs’ Energy Modes App note – AN0007”
Available on D2L as well as in Simplicity

EFM32 CMU application note - AN0004
Available on D2L as well as in Simplicity

EFM32 GPIO application note - AN0012
Available on D2L as well as in Simplicity

EFM32 Low Energy Timer LETIMER application note - AN0026
Available on D2L as well as in Simplicity

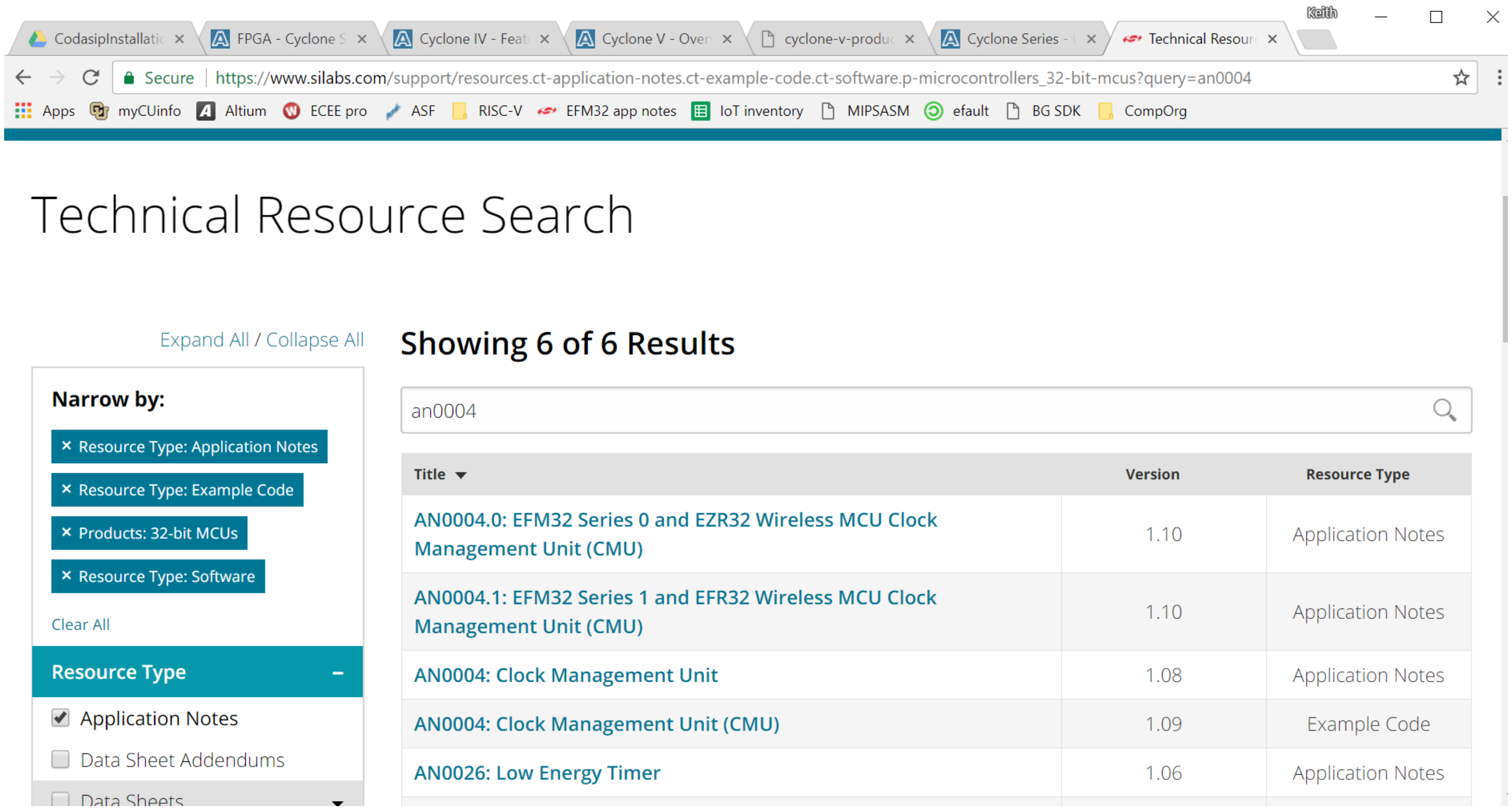
Important web link below. It will take you to the Silicon Labs’ Hardware Abstraction Layer home page for the Silicon Labs’ EFR32BG13 family of products:

http://devtools.silabs.com/dl/documentation/doxygen/5.3.2/

http://www.silabs.com/products/mcu/Pages/32-bit-mcu-application-notes.aspx

Application notes

+ Software examples



Class Survey Results

- I will use the results as a representative of the entire class.
- % are little or no experience
- What's is your experience in developing code and/or implementing an application for the following RF protocol?
  - WiFi – 92%
  - Bluetooth Classic – 72%
  - Bluetooth Smart – 92%
  - Bluetooth Mesh – 100%
- Rate your ability in using the integrated debugger of a micro controller IDE – 25%

Class Survey Results

- Rate your ability in “coding to the metal” on a micro-controller – 29%
- Rate your ability in programming an I2C device driver – 71%
- Rate your ability in using an I2C bus analyzer – 83%
- What are you wanting to get out of this course?
  - Energy efficient programs
  - OTA
  - Built small IoT systems
  - Wireless security
  - Coding to the metal

Quiz 1 review

Mesh networking advantages arise from the use of relaying messages from one point to another via hopping across bi-directional channels instead of using a [answer] device communicating with individual peripherals devices.

Quiz 1 review

At what input voltage would the Blue Gecko operate the most efficient?

- 2.5v
- answers 1, 2, and 3
- 3.3v
- 1.5
- answers 1, 2, 3, and 4
- 2.0v

Quiz 1 review

Which system would require the slowest response time?

- Smoke detector
- Automated plant waterer
- Space rocket motor controller
- Motion sensor

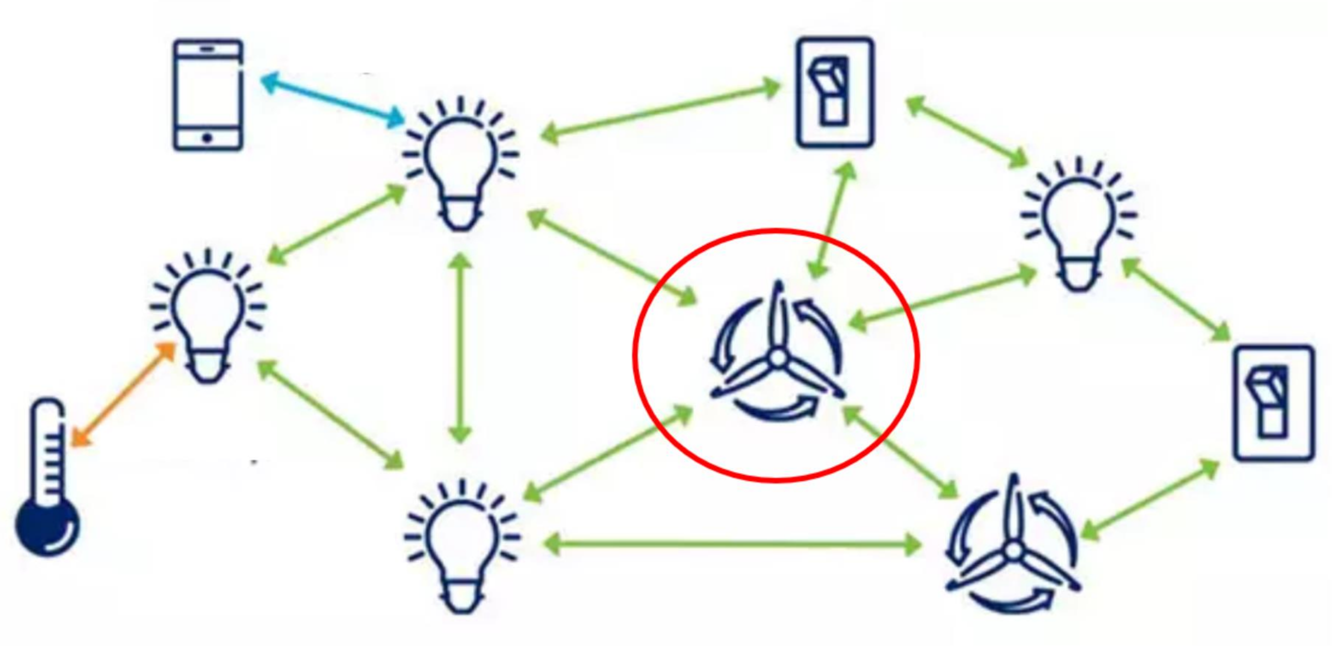
Quiz 1 review

Mesh networking is a key requirement for low-power wireless in commercial installation due to all the following? (select all that apply)

- [ ] standard
- [ ] scalability
- [ ] robustness
- [ ] low-power
- [ ] range

Quiz 1 review

For the device highlighted by the red circle, select all the Bluetooth Node type(s) that apply.



- Relay Node
- Proxy Node
- Friend Node
- Client
- Low Power Node
- Server

Quiz 1 review

A conventional program maintains ______ of the processing sequence from the beginning to the end. In contrast, event-driven gains control only ______ when handling events.

Quiz 1 review

Match the application to whether its requirements match more of a consumer or industrial IoT device.

Exercise watch	[Choose] ▾
Insulin pump	[Choose] ▾
Home heater	[Choose] ▾
Solar panel inverter	[Choose] ▾
Roof mounted personal weather station	[Choose] ▾

Quiz 1 review

In which Energy Mode of the Blue Gecko are the High-Frequency peripherals available?

(All of the particular peripheral must be available in that Energy Mode)

- [ ] EM3
- [ ] EM2
- [ ] EM4
- [ ] EM0
- [ ] EM1

Quiz 1 review

DMA reduces energy in a system by ____________ some memory transfers from the CPU.

Quiz 1 review

At what input voltage would the Blue Gecko operate the most efficient?

- answers 1, 2, and 3
- 2.5v
- 3.3v
- 1.5
- answers 1, 2, 3, and 4
- 2.0v

Quiz 1 review

[ ] is the length of time taken by a system to react to a give stimulus or event.

Quiz 1 review

To optimize energy savings, the microcontroller should [ ] as deeply, and

[ ] as seldom as possible.

Quiz 1 review

Match the Bluetooth Mesh node with its definition.

Remains awake and caches messages for another node	[Choose]
Receives and forward packets	[Choose]
Normally will be powered down for long periods of time	[Choose]
Enables connection to Bluetooth Smart devices	[Choose]

Quiz 1 review

For a GPIO pin not being utilized, what state should it be placed in to minimize power / energy?

- Input
- Wired-AND
- Output - push/pull
- Disabled

Quiz 1 review

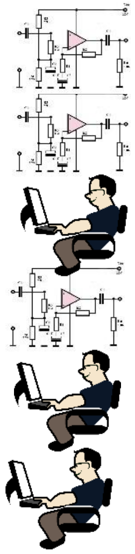
Which type of power is the cause that the faster the Blue Gecko operates to complete a task the more efficient the solution?

- Static power
- Dynamic power

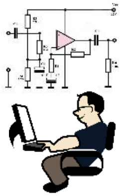
Energy or Current Monitors

- Energy or Current Monitors provide real time data on the use of energy
- How can these monitors be used to determine which routines consume?
  - Code correlation with the monitor enables the firmware engineer to pinpoint where the code is spending energy
  - And, thus, focus attention to reduce energy in those routines
- Verifies the energy efficiency of the firmware design

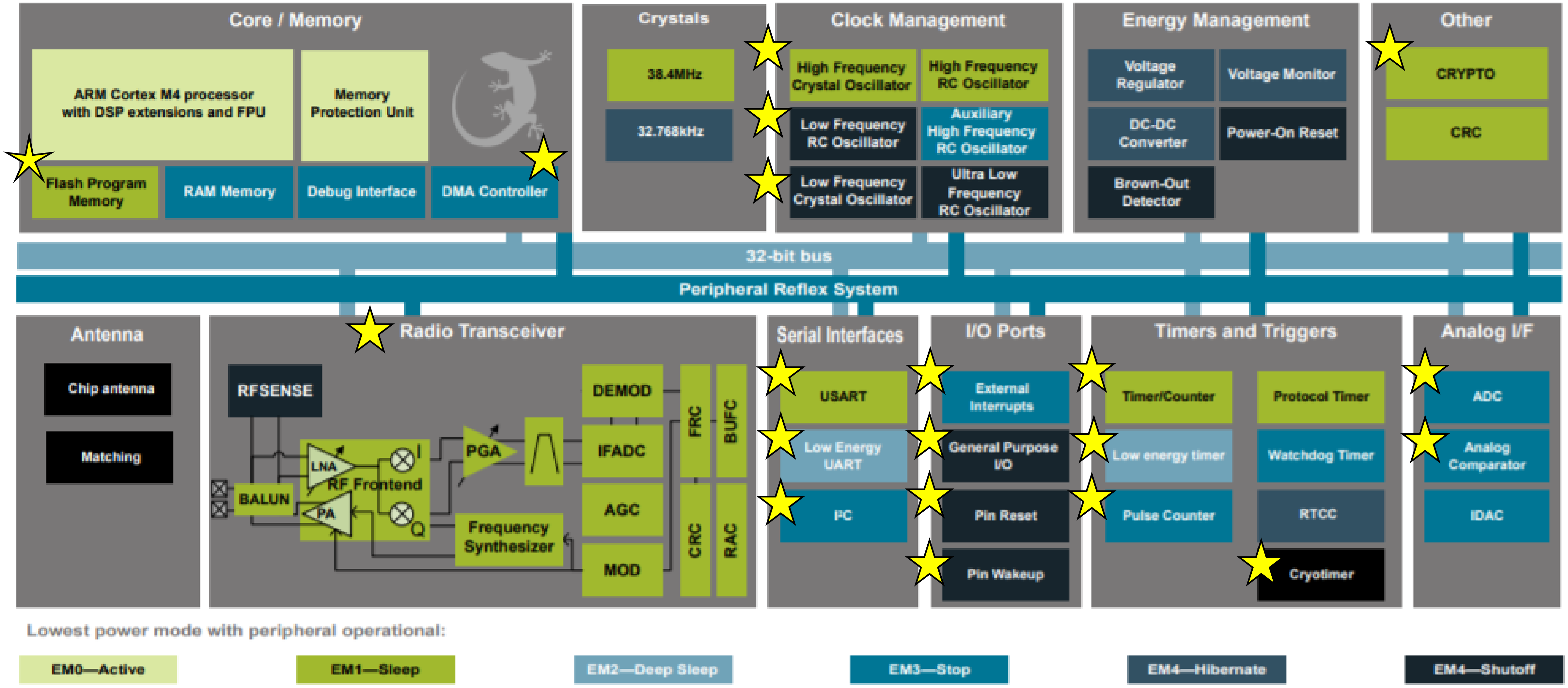
What are the characteristics that the firmware engineer can take advantage?



- Low Energy Microcontroller characteristics:
  - Higher Computational CPU
  - Very Low Active Power Consumption
  - Ultra Low Power Sleep Modes
  - Fast wake up times from sleep or low energy modes
  - Autonomous Peripherals
- Advanced autonomous peripheral functions:
  - Passive sensor state machines - LESENSE
  - Peripheral Intercommunication - PRS



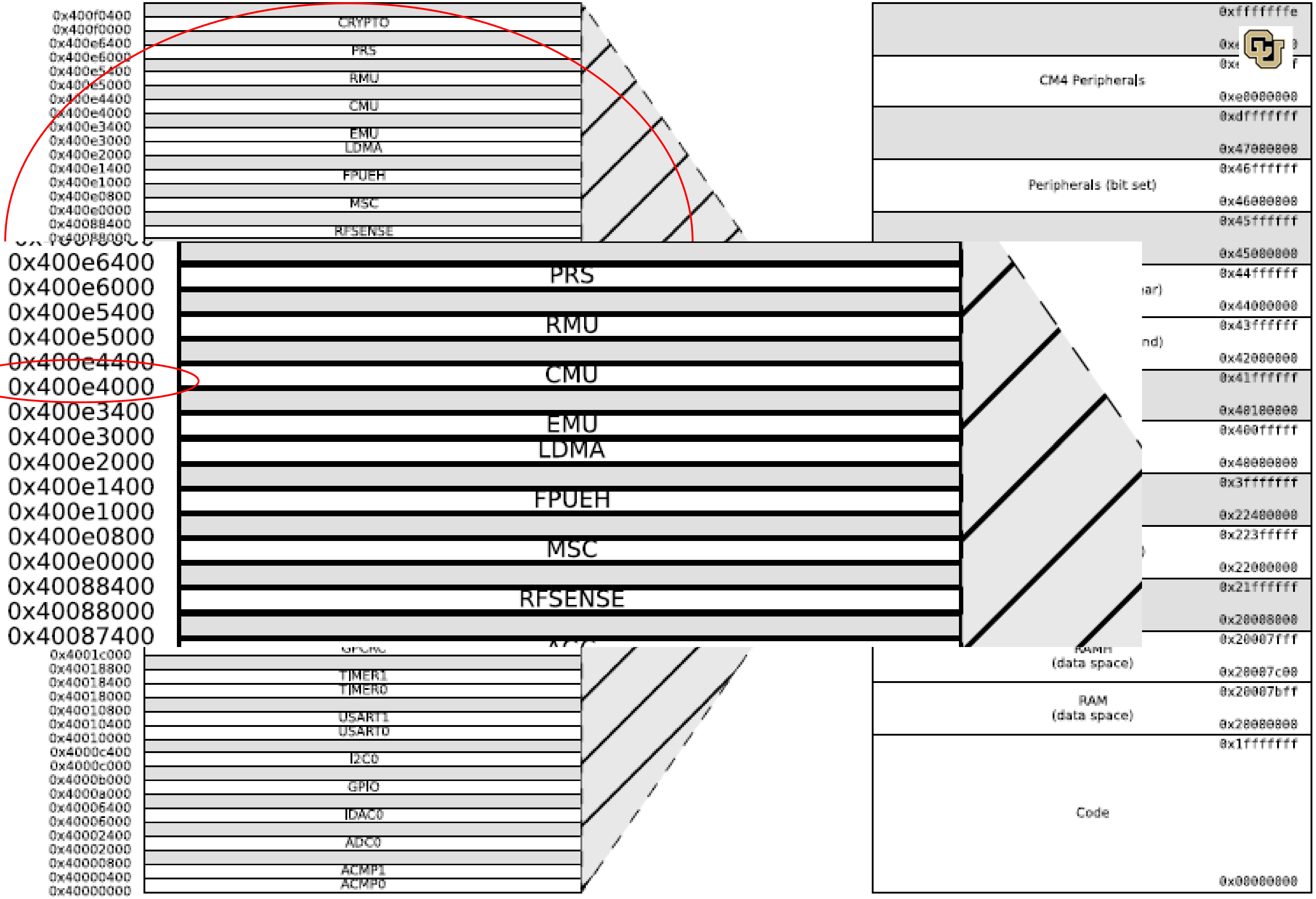
- Well architected Energy Modes
- Energy or Current monitors


Core / Memory
ARM Cortex M4 processor with DSP extensions and FPU
Memory Protection Unit
Flash Program Memory
RAM Memory
Debug Interface
DMA Controller
Crystals
38.4MHz
32.768kHz
Clock Management
High Frequency Crystal Oscillator
High Frequency RC Oscillator
Low Frequency RC Oscillator
Auxiliary High Frequency RC Oscillator
Low Frequency Crystal Oscillator
Ultra Low Frequency RC Oscillator
Energy Management
Voltage Regulator
Voltage Monitor
DC-DC Converter
Power-On Reset
Brown-Out Detector
Other
CRYPTO
CRC
32-bit bus
Peripheral Reflex System
Antenna
Chip antenna
Matching
Radio Transceiver
RFSENSE
DEMOD
IFADC
AGC
MOD
FRC
BUFC
CRC
RAC
PGA
LNA
RF Frontend
BALUN
PA
I
Q
Frequency Synthesizer
Serial Interfaces
USART
Low Energy UART
I²C
I/O Ports
External Interrupts
General Purpose I/O
Pin Reset
Pin Wakeup
Timers and Triggers
Timer/Counter
Protocol Timer
Low energy timer
Watchdog Timer
Pulse Counter
RTCC
Cryotimer
Analog I/F
ADC
Analog Comparator
IDAC
Lowest power mode with peripheral operational:
EM0—Active
EM1—Sleep
EM2—Deep Sleep
EM3—Stop
EM4—Hibernate
EM4—Shutoff

Accessing registers directly using C-code

CMU base address is:

0x400e4000



Accessing registers directly using C-code

CMU Interrupt Enable register offset address 0x0AC

0x024	CMU_HFXOCTRL	RW	HFXO Control Register
0x028	CMU_HFXOCTRL1	RW	HFXO Control 1
0x02C	CMU_HFXOSTARTUPCTRL	RW	HFXO Startup Control
0x030	CMU_HFXOSTEADYSTATECTRL	RW	HFXO Steady State control
0x034	CMU_HFXOTIMEOUTCTRL	RW	HFXO Timeout Control
0x038	CMU_LFXOCTRL	RW	LFXO Control Register
0x050	CMU_CALCTRL	RW	Calibration Control Register
0x054	CMU_CALCNT	RWH	Calibration Counter Register
0x060	CMU_OSCENCMD	W1	Oscillator Enable/Disable Command Register
0x064	CMU_CMD	W1	Command Register
0x070	CMU_DBGCLKSEL	RW	Debug Trace Clock Select
0x074	CMU_HFCLKSEL	W1	High Frequency Clock Select Command Register
0x080	CMU_LFACLKSEL	RW	Low Frequency A Clock Select Register
0x084	CMU_LFBCLKSEL	RW	Low Frequency B Clock Select Register
0x088	CMU_LFECLKSEL	RW	Low Frequency E Clock Select Register
0x090	CMU_STATUS	R	Status Register
0x094	CMU_HFCLKSTATUS	R	HFCLK Status Register
0x09C	CMU_HFXOTRIMSTATUS	R	HFXO Trim Status
0x0A0	CMU_IF	R	Interrupt Flag Register
0x0A4	CMU_IFS	W1	Interrupt Flag Set Register
0x0A8	CMU_IFC	(R)W1	Interrupt Flag Clear Register
0x0AC	CMU_IEN	RW	Interrupt Enable Register

Accessing registers directly using C-code

- CMU->IEN
  - CMU base address 0x400e4000
  - IEN register offset + 0x000000aC
  - CMU->IEN = 0x400e40aC

Accessing a register bit using C-code

- HFXO ready bit is bit 1
- LFXO ready bit is bit 3

- Register name bit equals a 1 in the correct bit position

- CMU_IEN_HFXORDY is
  - 0b00000010
- CMU_IEN_LFXORDY is
  - 0b00001000

12.5.26 CMU_IEN - Interrupt Enable Register

Offset	Bit Position																															
0x0AC	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	0	0	0	0	0		0	0	0	0	0	0	0
Access	RW																	RW	RW	RW	RW	RW	RW	RW		RW	RW	RW	RW	RW	RW	RW
Name	CMUERR																	LFTIMEOUTERR	HFRCODIS	HFXOSHUNTOPTRDY	HFXOPEAKDETRDY	HFXOPEAKDETERR	HFXOAUTOSW	HFXODISERR		CALOF	CALRDY	AUXHFRCORDY	LFXORDY	LFRCORDY	HFXORDY	HFRCORDY

Accessing registers directly using C-code

- CMU->IEN
  - CMU base address 0x400e4000
  - IEN register offset + 0x000000aC
  - CMU->IEN = 0x400e40aC
- CMU_IEN_HFXORDY | CMU_IEN_LFXORDY
  - CMU_IEN_HFXORDY 0b00000010
  - CMU_IEN_LFXORDY 0b00001000
  - CMU_IEN_HFXORDY | CMU_IEN_LFXORDY 0b00001010
- CMU->IEN = CMU_IEN_HFXORDY | CMU_IEN_LFXORDY;
  - CMU Interrupt Enable Register value at 0x400e40aC now is 0b00001010

Not clearing bits when you are adding setting a bit in a register

- Assume LETIMER0 already has the UF bit set in the Interrupt Enable Register
  - LETIMER0 currently is set to 0x00000004 (UF bit set)
- Now, lets also enable the COMP1 interrupt to the LETIMER0 peripheral
  - LETIMER0->IEN = LETIMER_IEN_COMP1;
    - LETIMER0 now is set to 0x00000002 (Overwrote UF interrupt bit)
- These are correct syntaxes to add a bit and not overwrite a bit
  - LETIMER0->IEN = LETIMER0->IEN | LETIMER_IEN_COMP1;
  - LETIMER0->IEN |= LETIMER_IEN_COMP1;
  - LETIMER_IntEnables(LETIMER0, LETIMER_IEN_COMP1);

```
__STATIC_INLINE void LETIMER_IntEnable(LETIMER_TypeDef *letimer, uint32_t flags)
{
  letimer->IEN |= flags;
}
```

Clearing a single bit(s) without disturbing others in a register

- Assume LETIMER0 already has both the COMP1 & UF bits set in the Interrupt Enable Register
  - LETIMER0 currently is set to 0x00000006 (UF & COMP1 bits set)
- Now, lets disable the UF interrupt to the LETIMER0 peripheral
  - LETIMER0->IEN = ~LETIMER_IEN_UF;
    - LETIMER0 now is set to 0xFFFFFFFB (Overwrote all bits)
- These are correct syntaxes to add a bit and not overwrite a bit
  - LETIMER0->IEN = LETIMER0->IEN & ~LETIMER_IEN_UF;
  - LETIMER0->IEN &= ~LETIMER_IEN_UF;
  - LETIMER_IntDisables(LETIMER0, LETIMER_IEN_UF);

```
__STATIC_INLINE void LETIMER_IntDisable(LETIMER_TypeDef *letimer, uint32_t flags)
{
  letimer->IEN &= ~flags;
}
```

Low Energy

$$\text{Energy} = \text{Power} \times \text{Time}$$

- Battery Capacity is measured in Energy such as mA-h
- To increase Batter Life, Energy must be reduced
  - Decrease Power
  - Decrease Time
- Low Energy firmware design minimizes both Power and Time

CU RISC-V processor data

Processor	A	B	C	D
Cycles/loop	2,191,817	109,181	68,211	29,200
Area	39,673	49,289	44,845	242,997
Power/cycle	143,347	145,247	144,288	492,895

Which processor is the higher performing processor?

What additional information required?

CU RISC-V processor data

Processor	A	B	C	D
	75	75	25	75
Cycles/loop	2,191,817	109,181	68,211	29,200
Area	39,673	49,289	44,845	242,997
Power/cycle	143,347	145,247	144,288	492,895

Which processor is the higher performing processor?

CU RISC-V processor data

Processor	A	B	C	D
	75	75	25	75
Cycles/loop	2,191,817	109,181	68,211	29,200
Area	39,673	49,289	44,845	242,997
Power/cycle	143,347	145,247	144,288	492,895

Which processor is the lower power processor?

Performance (uS)	29224	1456	2728	389
	100%	2008%	1071%	7506%

CU RISC-V processor data

Processor	A	B	C	D
	75	75	25	75
Cycles/loop	2,191,817	109,181	68,211	29,200
Area	39,673	49,289	44,845	242,997
Power/cycle	143,347	145,247	144,288	492,895
Power per loop	314,190,391,499	15,858,212,707	9,842,028,768	14,392,534,000
	100%	5%	3%	5%
Performance (uS)	29224	1456	2728	389
	100%	2008%	1071%	7506%

Enabling a peripheral clock

- Example: Enabling the clock to the I2C0 peripheral
- Emlib routine:
  - **void, CMU_ClockEnable(CMU_Clock_TypeDef clock, bool enable)**
  - CMU_Clock_TypeDef

cmuClock_ACMP0	Analog comparator 0 clock.
cmuClock_ACMP1	Analog comparator 1 clock.
cmuClock_IDAC0	Digital to analog converter 0 clock.
cmuClock_ADC0	Analog to digital converter 0 clock.
cmuClock_I2C0	I2C 0 clock.
cmuClock_CORE	Core clock
cmuClock_LFA	Low frequency A clock

  - **Bool**
    - **true** to enable clocking to the I2C0
  - CMU_ClockEnable(cmuClock_I2C0, true);

Enabling a peripheral clock

- Direct register access

12.5.28 CMU_HFPERCLKEN0 - High Frequency Peripheral Clock Enable Register 0

Offset	Bit Position																															
0x0C0	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	0	0	0	0	0	0	0
Access																							RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Name																							IDAC0	ADC0	I2C0	CRYOTIMER	ACMP1	ACMP0	USART1	USART0	TIMER1	TIMER0

- CMU->HFPERCLKEN0 = CMU->HFPERCLKEN0 | CMU_HFPERCLKEN0_I2C0;
- Or,
- CMU->HFPERCLKEN0 |= CMU_HFPERCLKEN0_I2C0;
- Example to disable:
  - CMU->HFPERCLKEN0 &= ~CMU_HFPERCLKEN0_I2C0;

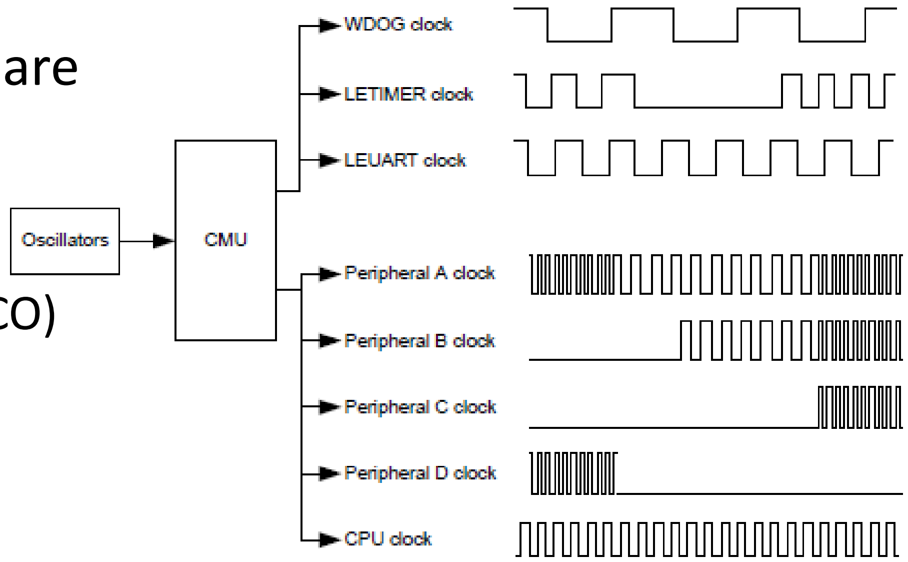

OTP Authentication using BLE
0:02 / 4:04

Ideal CMOS FET

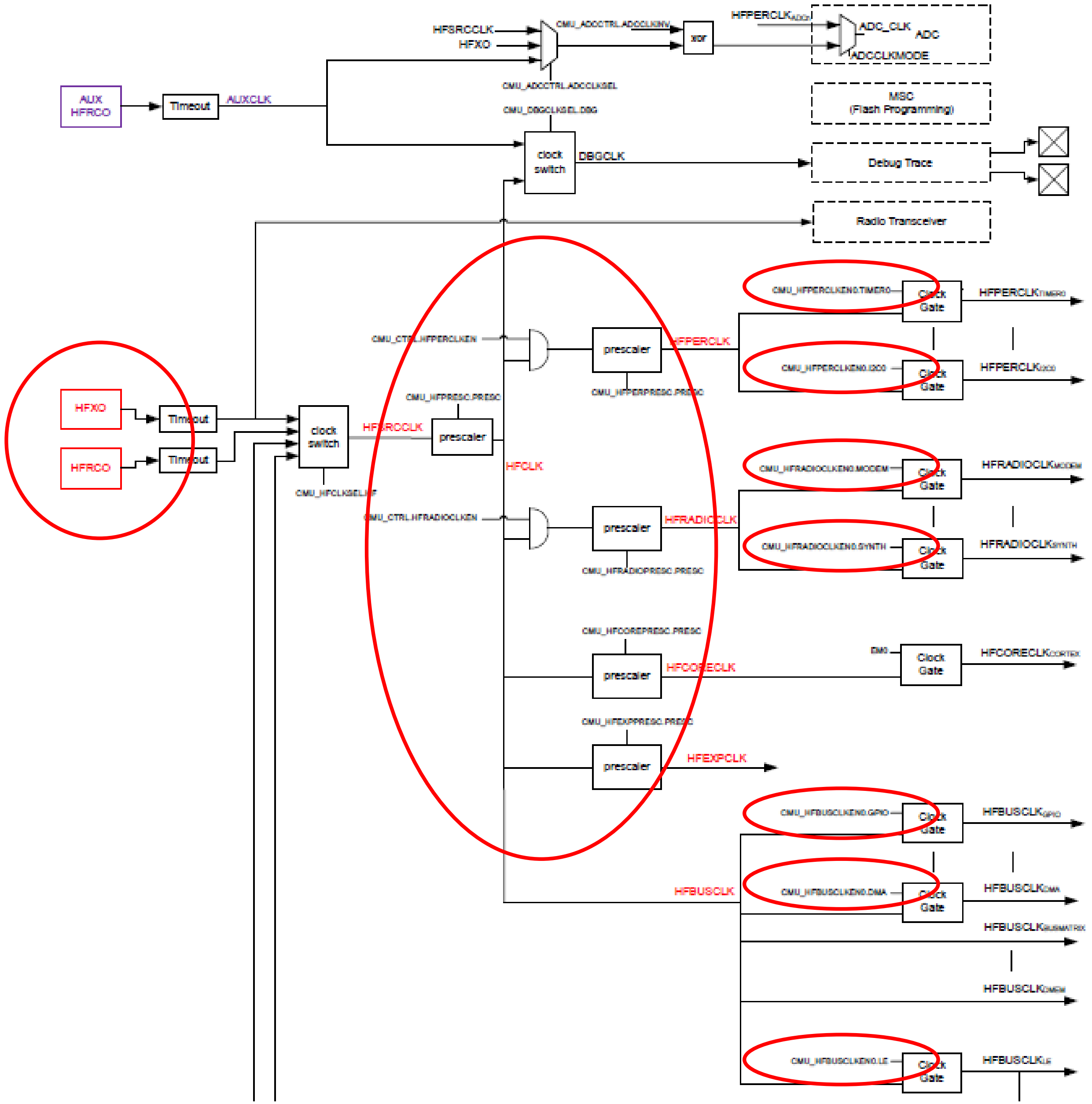
- CMOS logic reduces power consumption because no current flows (ideally), and thus no power is consumed, except when the inputs to logic gates are being switched. CMOS accomplishes this current reduction by complementing every nMOSFET with a pMOSFET and connecting both gates and both drains together.
- What does the above imply if you want to minimize energy?
  - CMOS logic is not clocked or switched
  - Lower switching frequency will result in reduced current/power/energy
    - Caveat: Only if it does not extend the time the CPU remains in EM0

Clock tree assumptions

- Goal is low power / low energy applications
- To achieve this goal:
  - Clock sources of various frequencies, tolerances, and power are available
    - 38 MHz - 40 MHz High Frequency Crystal Oscillator (HFXO)
    - 1 MHz - 38 MHz High Frequency RC Oscillator (HFRCO)
    - 1 MHz - 38 MHz Auxiliary High Frequency RC Oscillator (AUXHFRCO)
    - 32768 Hz Low Frequency Crystal Oscillator (LFXO)
    - 32768 Hz Low Frequency RC Oscillator (LFRCO)
    - 1000 Hz Ultra Low Frequency RC Oscillator (ULFRCO)
  - Prescalers on each clock tree to optimize energy to required performance
  - Clock enables to only consume energy on peripherals required for the application
  - If it's a low energy micro, what would be the default clocking state?
    - By default, each peripheral clock is disabled / turned off

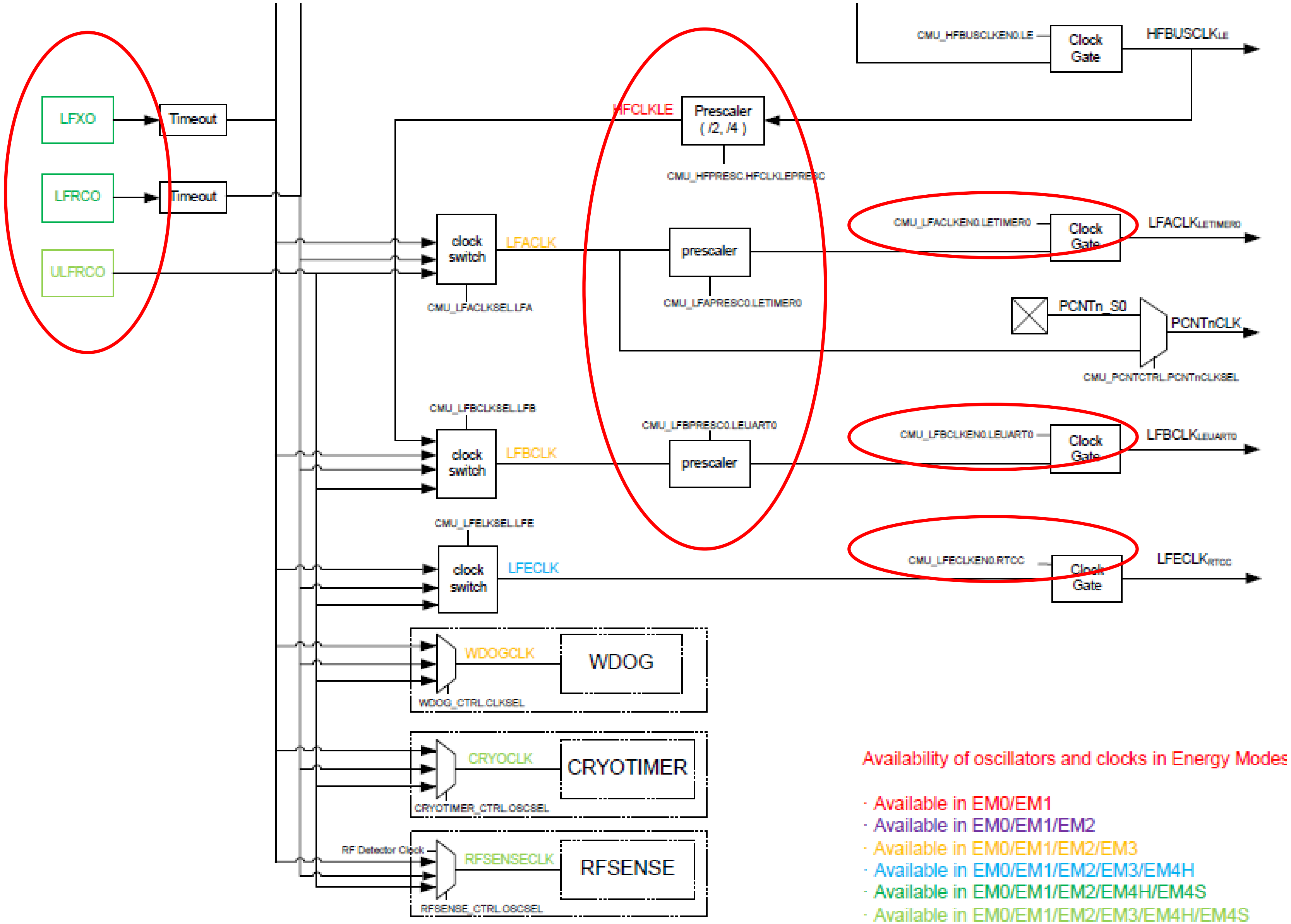


Blue Gecko High Frequency Clock Tree



Silicon Labs' EFR32BG1 Reference Manual

Blue Gecko High Frequency Clock Tree



CMU – Clock Management Unit

- The Clock Management Unit (CMU) is responsible for controlling the oscillators and clocks in the EFR32.
  - Provides the capability to turn on and off the clock on an individual basis to all peripheral modules
  - Enables/Disables and configures the available oscillators.
  - The high degree of flexibility enables software to minimize energy consumption in any specific application by not wasting power on peripherals and oscillators that do not need to be active.

IMPORTANT PROGRAMMING NOTE!!!

- If you disable both of the HF clock sources, HFXO and HFRCO, you will "brick" your dev kit

I know, I have done it!

- Before disabling the HFXO, insure that you first enable the HFRCO and select the HFRCO as the HF clock source!