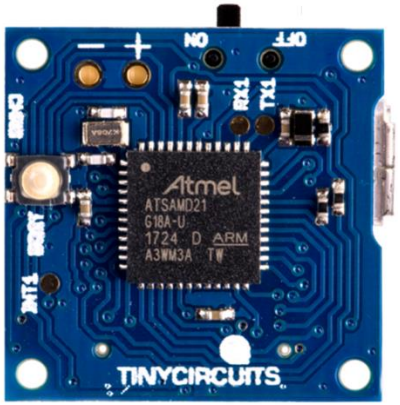
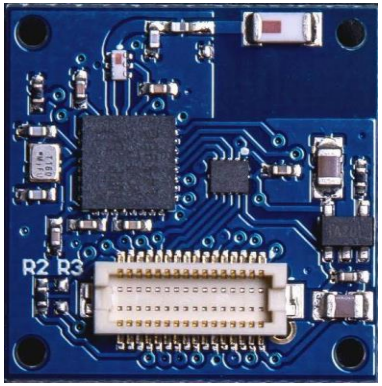
CSE190 Winter 2025



Lecture 5

GPIO and Time



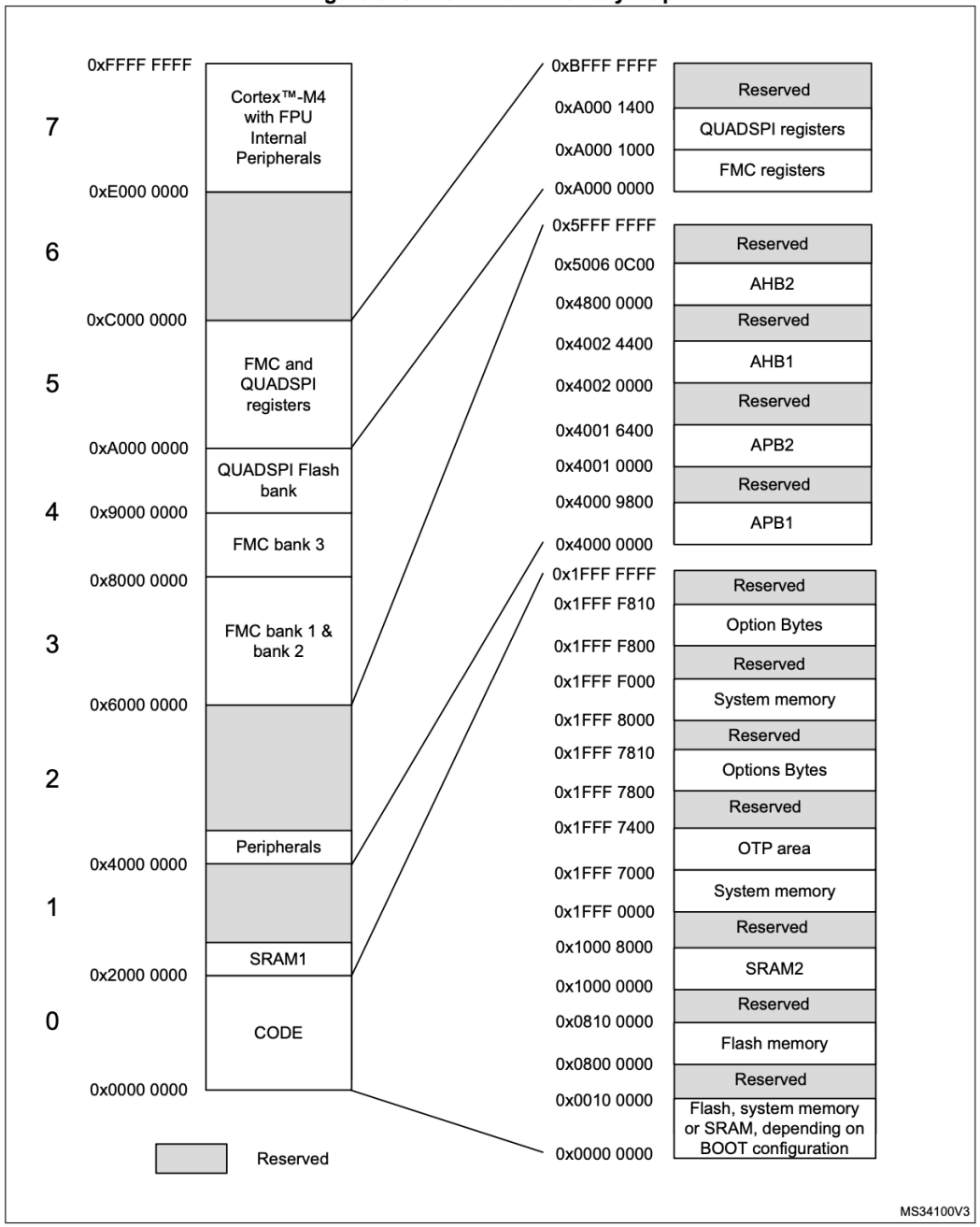
Wireless Embedded Systems

Aaron Schulman

Reading and writing with MMIO is not like talking to RAM

- MMIO reads and writes hardware device registers
- Read and write to registers can cause peripherals to begin or end an operation
  - Reading is not a passive operation; it can make the hardware to do something!
  - E.g., Read to clear an interrupt flag, or to advance
  - Writing often starts operations
    E.g., Send this data over the UART bus

Figure 8. STM32L475xx memory map



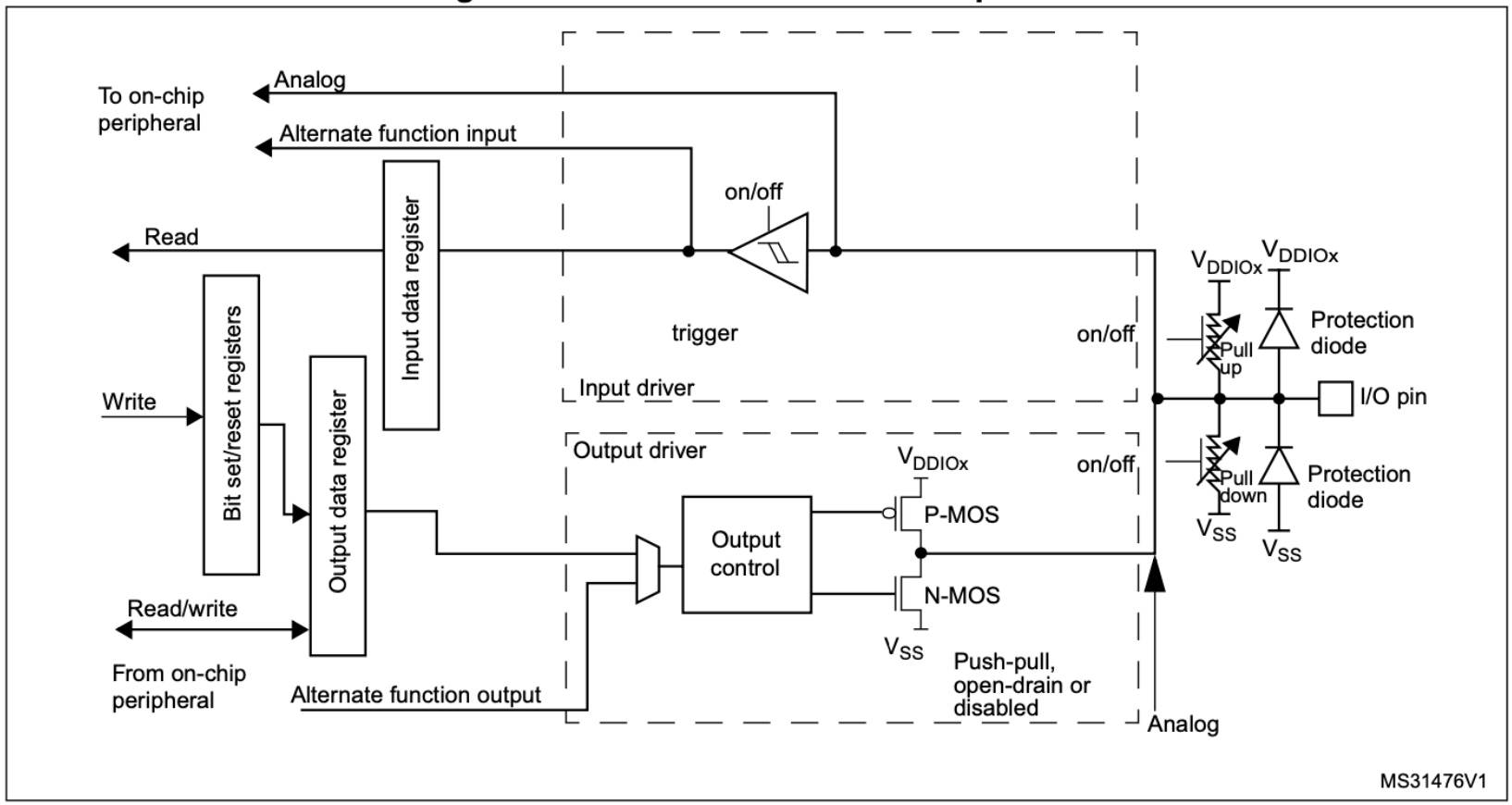
GPIOs are the general digital I/O device

Each GPIO pin represents one bit in memory: if the pin is on it's a 1, off it's a 0

That bit can be an input or an output

- GPIOs can be used to control lights (light on or off), but even more
- Indicating that an event just happened
  - Interrupt the radio to tell it to transmit data
  - Interrupt the CPU to tell it a button was pressed
  - Read pin status to receive configuration messages
- Debugging
  - Did this one part of my code actually execute?
  - Is the timer firing at the interval that I expect it to fire (connect GPIO to oscilloscope)?
  - Why using GPIO?
    - GPIO ops are lightweight

Figure 23. Basic structure of an I/O port bit



GPIO Output Configurations

Table 39. Port bit configuration table(1)

MODE(i) [1:0]	OTYPER(i)	OSPEED(i) [1:0]	PUPD(i) [1:0]		I/O configuration	
01	0	SPEED [1:0]	0	0	GP output	PP
	0		0	1	GP output	PP + PU
	0		1	0	GP output	PP + PD
	0		1	1	Reserved	
	1		0	0	GP output	OD
	1		0	1	GP output	OD + PU
	1		1	0	GP output	OD + PD
	1		1	1	Reserved (GP output OD)	
10	0	SPEED [1:0]	0	0	AF	PP
	0		0	1	AF	PP + PU
	0		1	0	AF	PP + PD
	0		1	1	Reserved	
	1		0	0	AF	OD
	1		0	1	AF	OD + PU
	1		1	0	AF	OD + PD
	1		1	1	Reserved	

GPIO Input configuration

Table 39. Port bit configuration table[1] (continued)

MODE(i) [1:0]	OTYPER(i)	OSPEED(i) [1:0]		PUPD(i) [1:0]		I/O configuration	
00	x	x	x	0	0	Input	Floating
	x	x	x	0	1	Input	PU
	x	x	x	1	0	Input	PD
	x	x	x	1	1	Reserved (input floating)	
11	x	x	x	0	0	Input/output	Analog
	x	x	x	0	1	Reserved	
	x	x	x	1	0		
	x	x	x	1	1		

1. GP = general-purpose, PP = push-pull, PU = pull-up, PD = pull-down, OD = open-drain, AF = alternate function.

What time is the Apple Watch tracking?

How often | *Granularity*

Clock (all the time | sec)
Alarm (all the time | sec)
Stopwatch (when open | msec)
Sync (all the time | sec)
UI (when open | msec)
Buzzer (when buzzing | msec)
WiFi (when communicating | usec)

Why do we need timers?

- In general, why do we need timers?
  - What time is it now?
  - How much time has elapsed since I last checked?
  - Let me know when this much time passes.
  - When did this external input occur?

What peripherals do we use to track time?

(all the time | sec) - [Alarm, Sync]

32-bit Real time clock (RTC) peripheral with interrupts

(when open/buzzing | msec) - [Stopwatch, UI, Buzzer]

Processor's *timer* peripheral with interrupts

(when communicating | usec) - [WiFi]

WiFi chip's internal timer peripheral with interrupts

What peripherals do we use to track time?

(all the time | sec) - [Alarm, Sync]

32-bit Real time clock (RTC) peripheral with interrupts

The term is used to avoid confusion with ordinary hardware clocks which are only signals that govern digital electronics, and do not count time in human units.

(when open/buzzing | msec) - [Stopwatch, UI, Buzzer]

Processor's *timer* peripheral with interrupts

(when communicating | usec) - [WiFi]

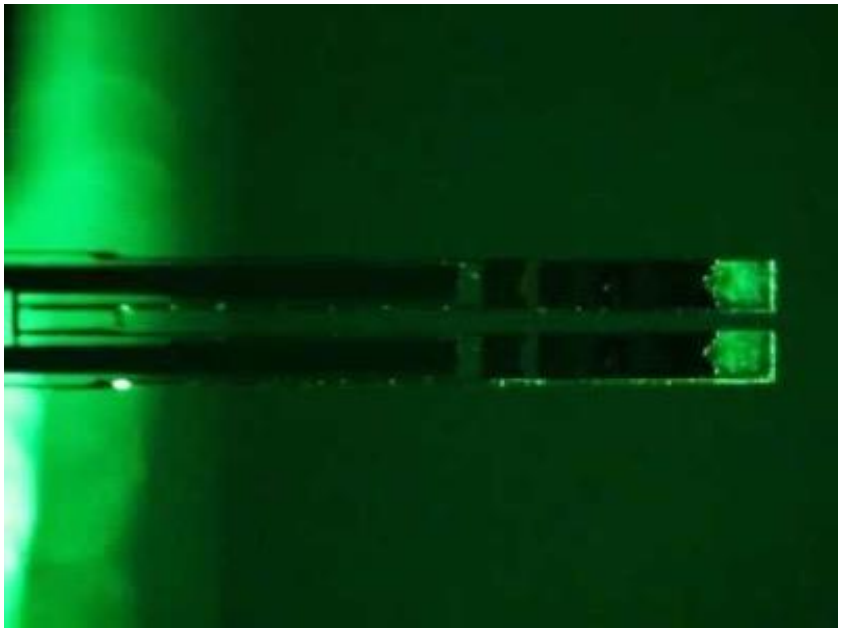
WiFi chip's internal timer peripheral with interrupts

Why do we need timers?

In the first project, what do we need timers for?

- Determining when to change LEDs
  - 20 Hz means change bits every 50 milliseconds
  - How to measure 50 ms?
  - Option 1: Use the timer hardware to let you know when 50 ms has passed.
  - Option 2: Count how many processor cycles it would take to equal 50 ms.

What is measuring the time?
Oscillators (generally, crystal oscillators)



Video: Crystal oscillators "go to war"