Kuwait University
College of Engineering and Petroleum

جامعة الكويت
KUWAIT UNIVERSITY

ME319 MECHATRONICS

Part I: The Brains – Microcontrollers, Software and Digital Logic

Lecture 6: Timers

Spring 2021

Ali AlSaibie

Lecture Plan

- Objectives:
  - Understand the fundamentals of a timer operation
  - Overview the application of Timers
  - Introduce Pulse Width Modulation

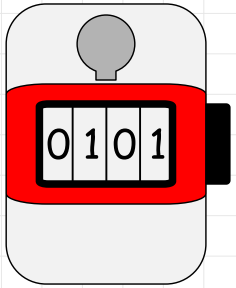
- Note: the timer operations reviewed in this lecture are based on the STM32F401x MCUs, but they largely apply to all MCU Timers. Names and terms may differ.

جامعة الكويت
KUWAIT UNIVERSITY

Timers – Mechanical Analogy

- This is a mechanical tick counter. Every time you click the middle button, it increments the counter by one.
- Continues to count **up** until 9999 then resets, mechanically, to 0000
- Now, replace the mechanical counter by a memory register, and
- Replace the button-clicking-action by an electric pulse
- You now have a digital timer.
- The size of the counter register (# of bits) and pulse rate:
  - Determine the speed of the count and the reset rate

Timers



Example 1

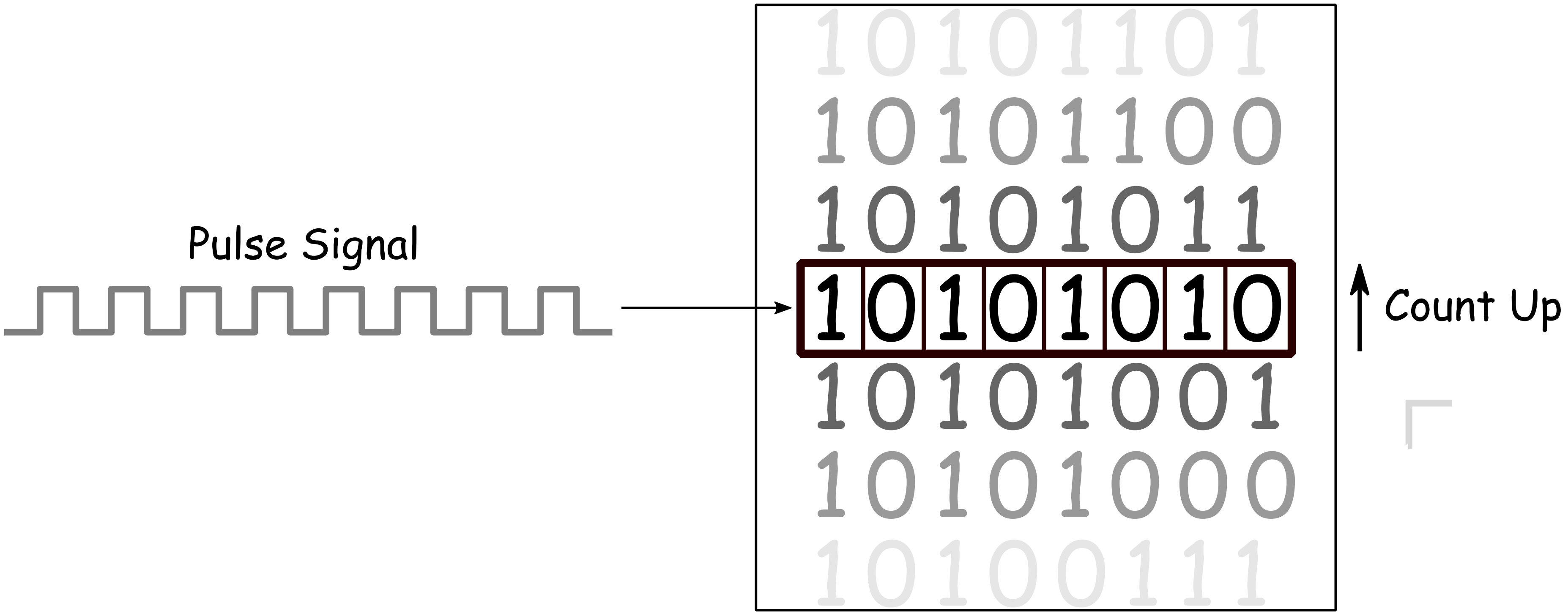
If you have an 8bit counter; a counter that counts from 0 to 255 then resets to 0. And you are counting up at a rate of 20Hz. How long does it take for you to count from 0 to 255?

جامعة الكويت
KUWAIT UNIVERSITY

- In the context of an MCU, a timer
- Counts **up** or **down**, either at a specific rate **or** in response to an event
- Timer count is stored in a memory register (8, 16, 32bit registers)
- A 16-bit timer will count **up** from 0x0000 to 0xFFFF, then rollover to 0x0000
  - Or can count **down** from 0xFFFF to 0x0000 and rollover to 0xFFFF
  - Can also be configured to rollover at a specific value (or start from one)
    - Can configure it to count to **0xF0E5**, for example, then rollover.
- When an MCU has multiple timer peripherals, usually:
  - Every timer peripheral can be independently set (rate, range, mode, direction)

جامعة الكويت
KUWAIT UNIVERSITY

- A timer increments (decrements) in response to a pulse
  - Rising Edge, Falling Edge, or Both
- The pulse can come from a clock source, which can be scaled (prescaler), or
- From an event signal (e.g. an external button press, encoder, modulated signal)



Example 2

We would like to have a timer rollover every 10ms. If the timer is running at 100MHz (count rate), what should be the starting count value (timer in count-down mode)

جامعة الكويت
KUWAIT UNIVERSITY

Timer (♪♪ What is it good for? ♪♪) Absolutely Everything

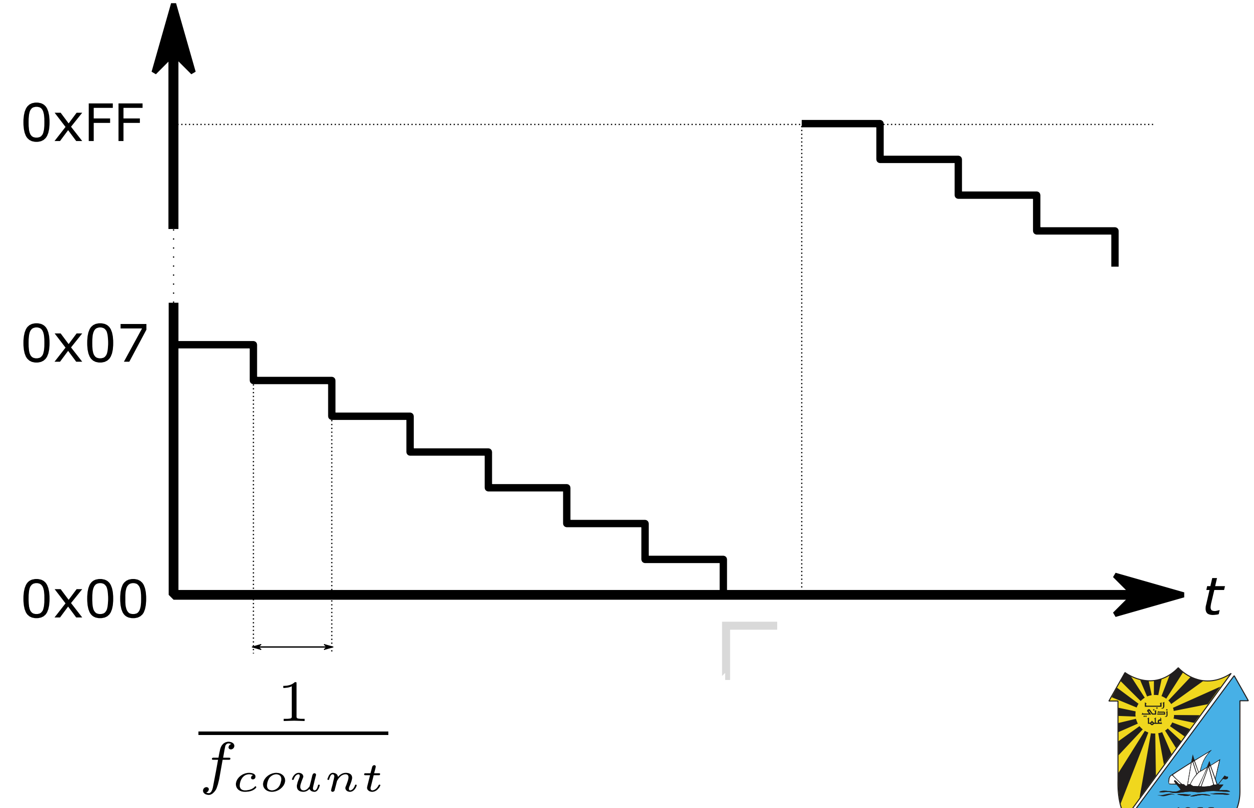
- Using timers, we can
- Keep track of elapsed time, or wait for a specific amount of time
  - *E.g. When you call delay(500); a timer peripheral is used*
- Call a function at a specific and deterministic rate
  - *E.g. Essential in applied control systems*
- Generate a signal with a specific frequency & on/off time ratio
  - *E.g. Generate a square wave, PWM or PPM signal*
- Record the time when an external or internal event occurred
  - *E.g. Register the frequency of a square wave signal*

Let's look at how each one is achieved.....

جامعة الكويت
KUWAIT UNIVERSITY

- An 8-bit count-down timer, can be configured to count **from** a maximum value of **0xFF.**
- Termed the **autoload value**, since the MCU will load it onto counter at rollover
- Rollover frequency:

$$f_{rollover} = \frac{f_{count}}{AutoLoad\,Value}$$



- Rollover period:

$$T_{rollover} = \frac{1}{f_{rollover}}$$
$$= T_{count} \times AutoLoad\,Value$$

جامعة الكويت
KUWAIT UNIVERSITY

- There are several **modes** a timer can be configured for

1. **Periodic Timer: Internal** MCU use
2. **Input Capture:** Records when an external **input** event occurred
3. **Output Compare:** Generates an external **output** waveform

Additional modes can be found on MCUs, but they are usually an extension of the above

In each of these modes, there is always a counter incrementing/decrementing at a specific rate, but their purpose and use are different.

Timers, just like other peripherals, operate independently from the CPU.

There is no CPU overhead from using timers; except for when

accessing timer data

Timer Modes – Periodic Timer

- When used as a general, or **periodic**, timer:
- General count up/down timer, counting at a set rate
- Can generate a timed interrupt **INT** (interrupt CPU at a specific rate), 1 usage:
  - Configure a timer (rate, count range) to issue an **INT** at every rollover
  - Then tie this **INT** to a specific function call.
  - *Now you have a periodic function call, use it to control a robot motor*
- Can also be checked, polled, by the program. 1 usage:
  - Configure a timer to run freely
  - If you want to *delay(500ms)*, wait for a number of counts then continue
  - # counts to wait are based on timer rate and delay duration requested

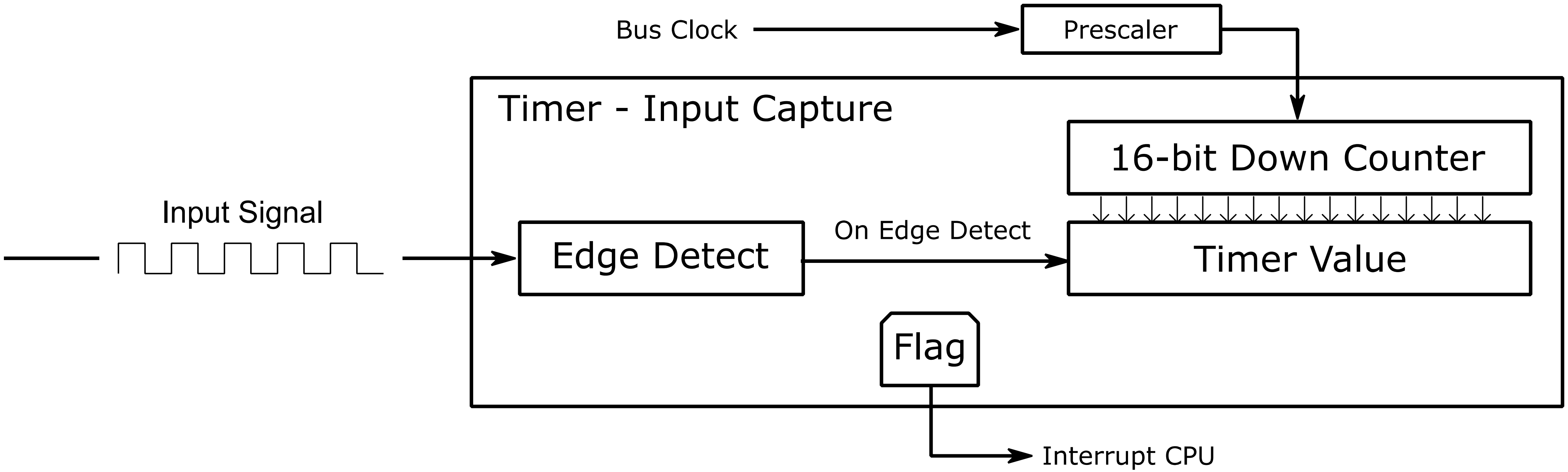
جامعة الكويت
KUWAIT UNIVERSITY

Timer Modes – Input Capture

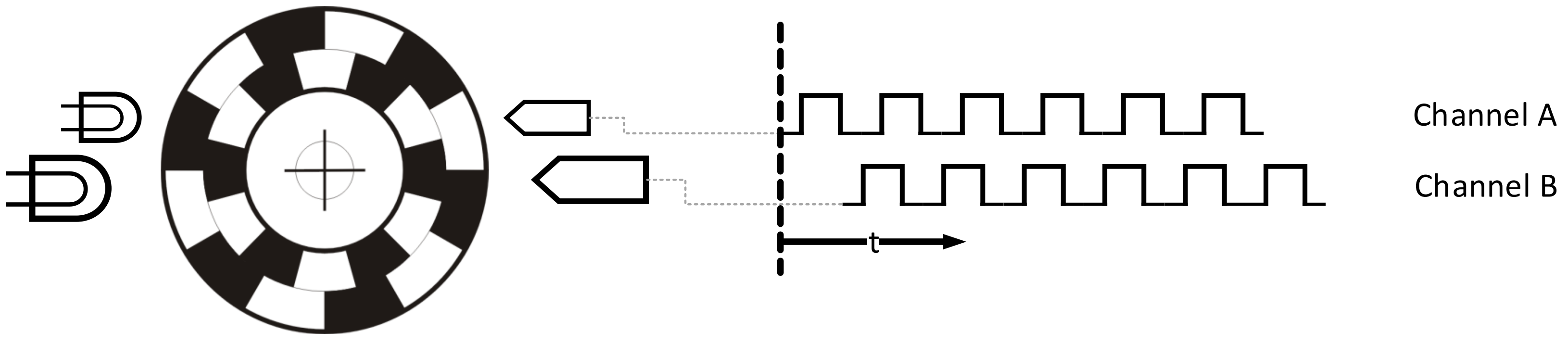
- In **Input** Capture mode, you have:
- A timer running with a certain configuration (rate, range, direction), **plus**
- Monitors for an external input event via a pin (Rising Edge, Falling or Both)
- When an event occurs, timer makes a **copy** of the count value in the timer register and stores it in a second memory register
- Can also issue an **INT** to CPU: "*Hey Boss, an event occurred, come down and read the count value for when it occurred.*"
- With this setup you can: measure frequency of an input signal
  - Record count values for two events, subtract difference and convert to time/frequency

جامعة الكويت
KUWAIT UNIVERSITY

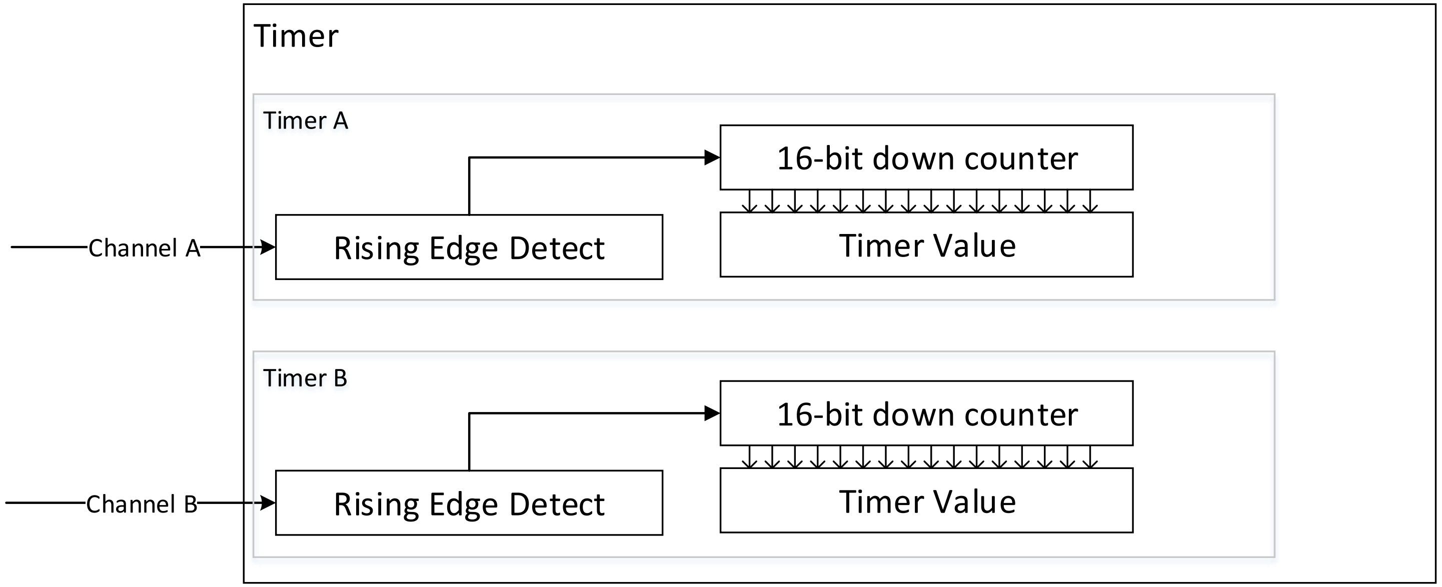
- Note that the timer counter keeps running uninterrupted, when event occurs a **snapshot** of the counter value is stored in a **different** register



- Time in Input Capture Mode is used with rotary encoders

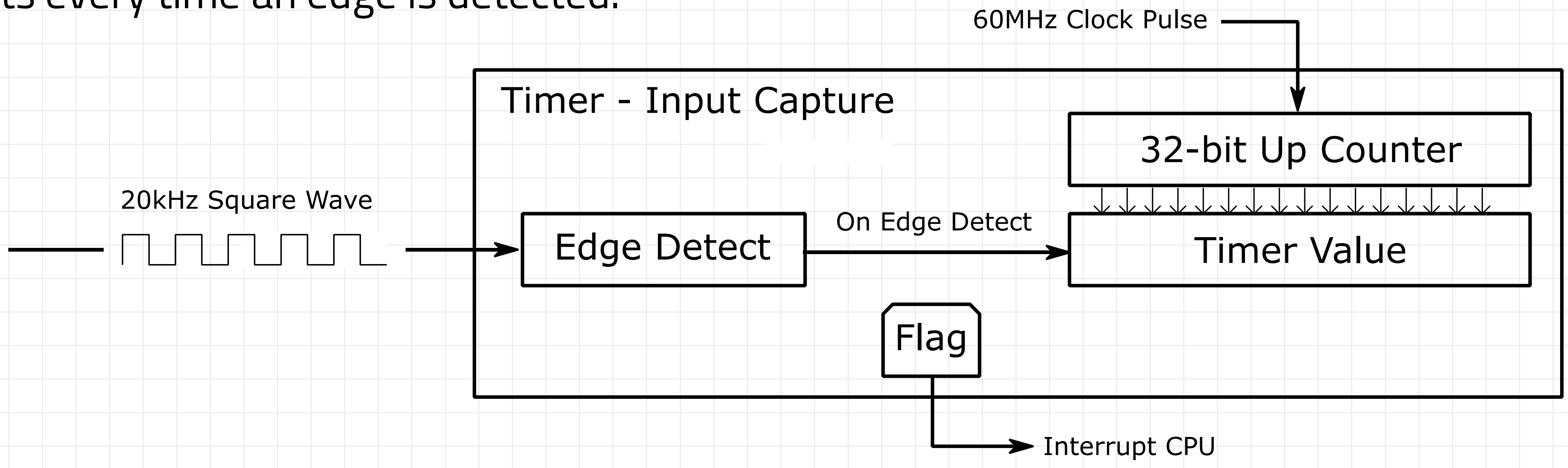


Channel A leads: Clockwise, otherwise CCW



Given the following Timer Configuration. If the detection mode is on Both Edges (Rising and Falling). What is the counter value when the edge detection occurs.

Assume the counter restarts every time an edge is detected.

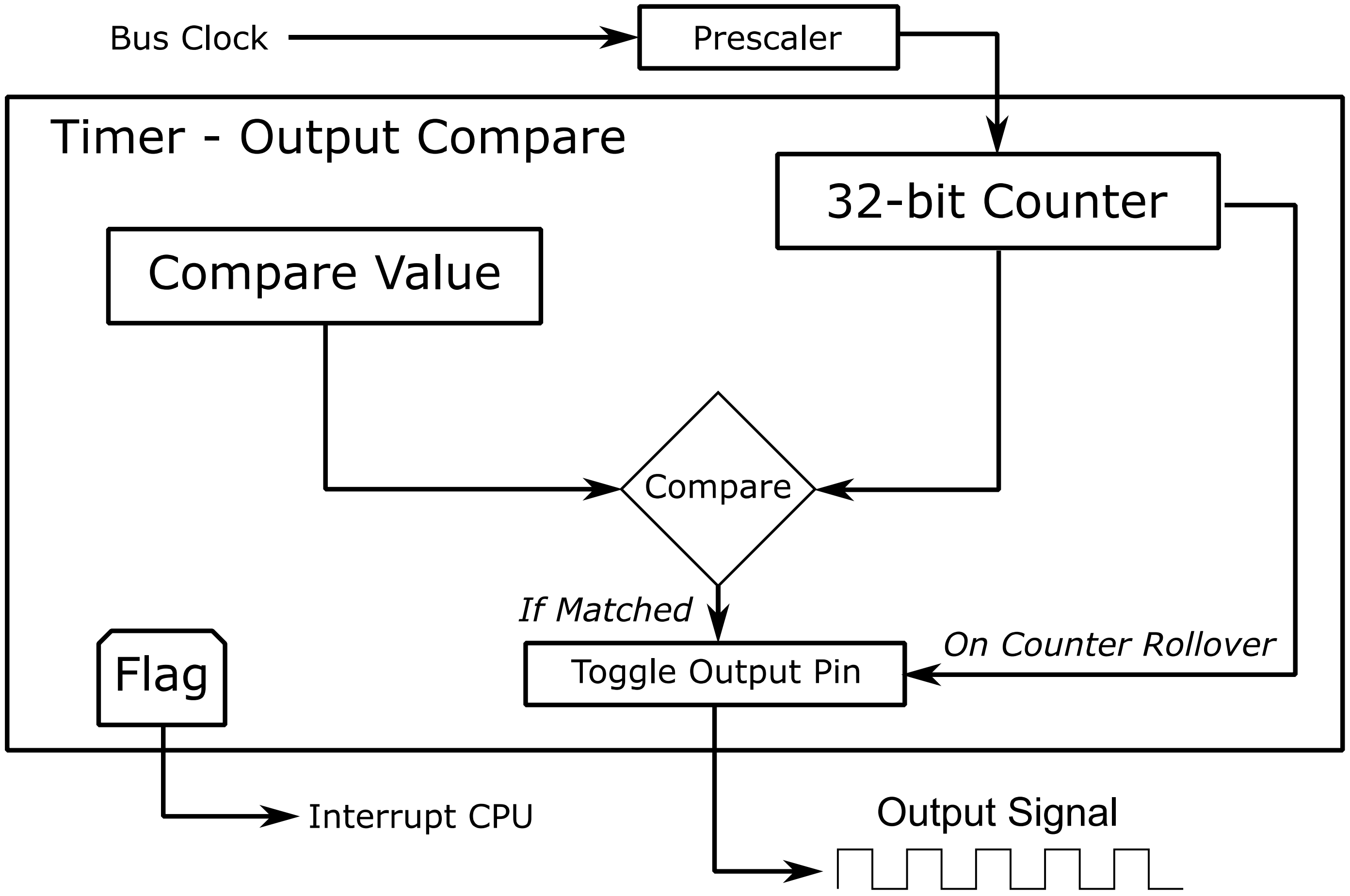


Timer Modes – Output Compare

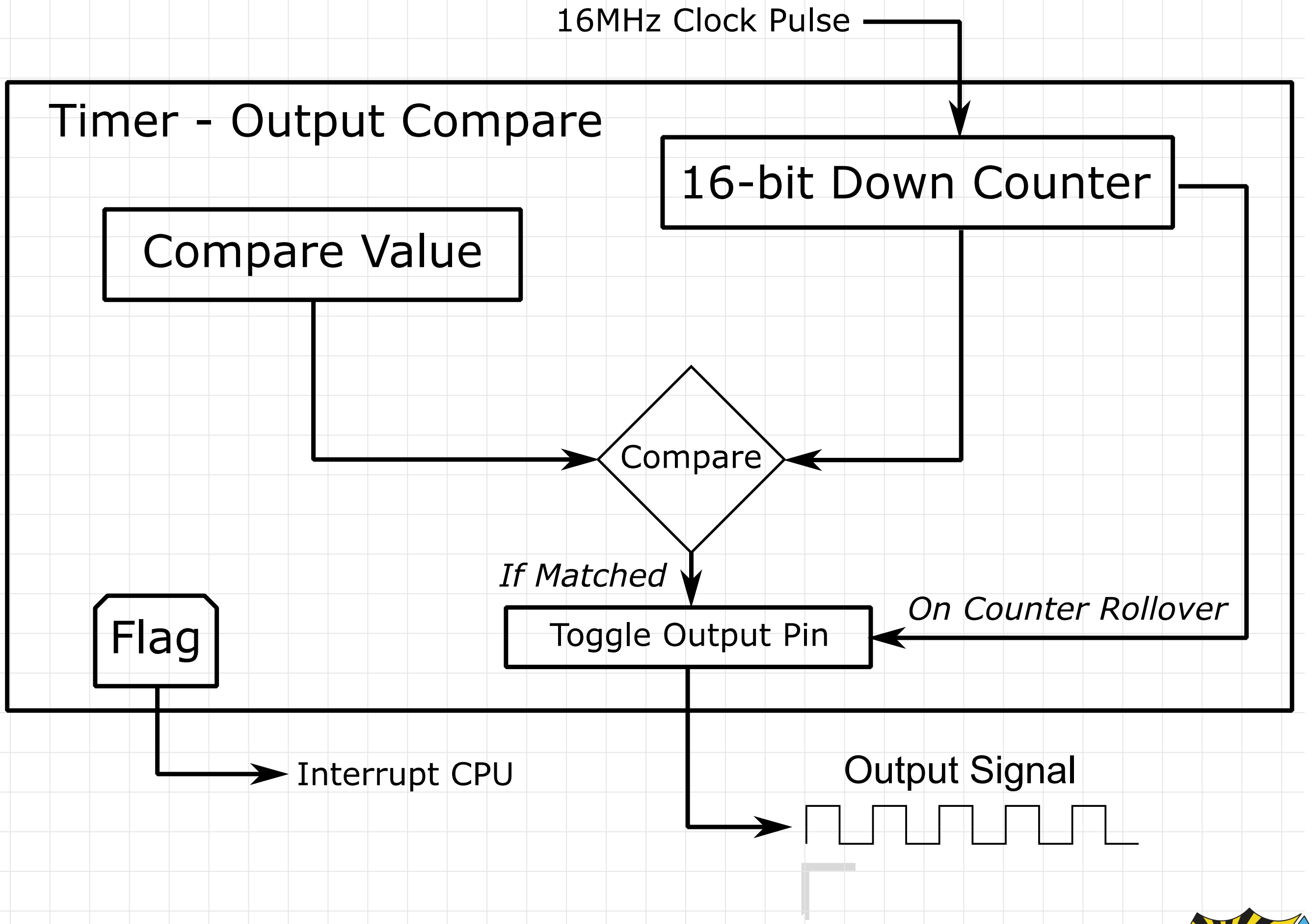
- In Output Compare, you have
- A timer running with a certain configuration (rate, range, direction), plus
- A fixed value to **compare** the count value of the timer with
- At the beginning of the count or at rollover, set a GPIO output pin
- If the counter reaches the **compare** value, clear the GPIO output pin.
- With this you can create a:
  - A variable width pulse (PWM)
    - Autoload Value determines signal freq, compare value determines duty cycle
  - A variable frequency signal (Square Wave).
    - Autoload Value determines signal freq:
      - compare value is $\frac{1}{2}$ autoload value

جامعة الكويت
KUWAIT UNIVERSITY

- By selecting the range (autoload value) in the counter, the count rate and the compare value an output signal can be designed and generated.



Given the following timer configuration. What should be the autoload value for the 16-bit down counter, and the compare value. In order to generate a 100kHz square wave signal.



جامعة الكويت
KUWAIT UNIVERSITY

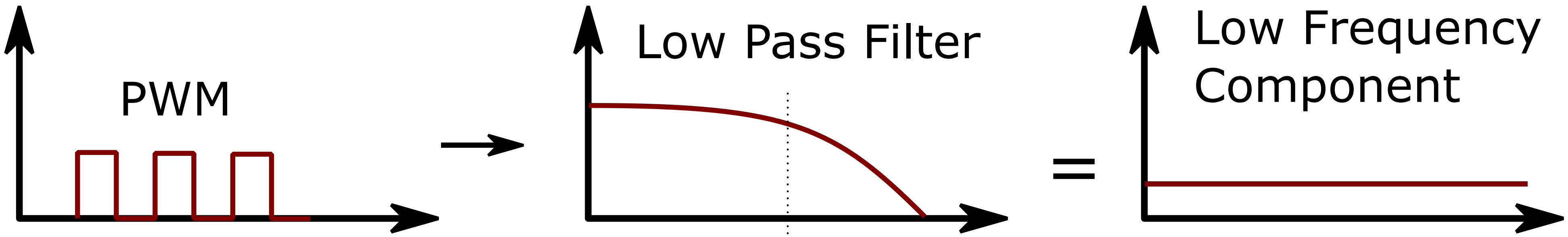
- When we have an analog input to the microcontroller, we can use the ADC to convert to digital, how about going the other way?
- Say we want to vary the output voltage to control the brightness of an LED, or the speed of a motor?
- Microcontrollers work in 1's and 0's, how to achieve a value in between?
- Options:
  - DAC: Digital to Analog Conversion (Computationally costly)
  - PWM: Pulse Width Modulation (Simple and easy)
- Many systems (electromechanical specifically) have a low pass filter characteristic to high frequency signals.
- They can accept a high frequency binary signal and average it

جامعة الكويت
KUWAIT UNIVERSITY

Pulse Width Modulation

- A variable signal can be generated with one output

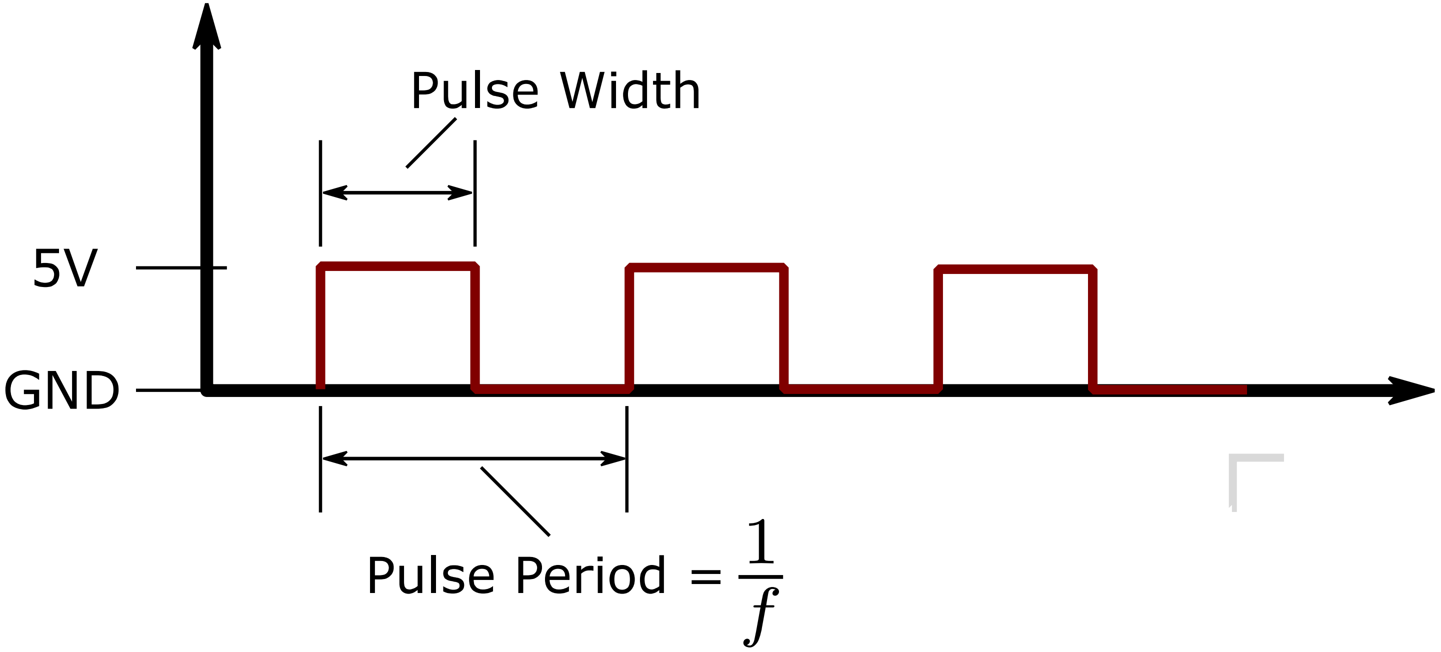
- In fact, a PWM with an external Low Pass Filter circuit can perform DAC



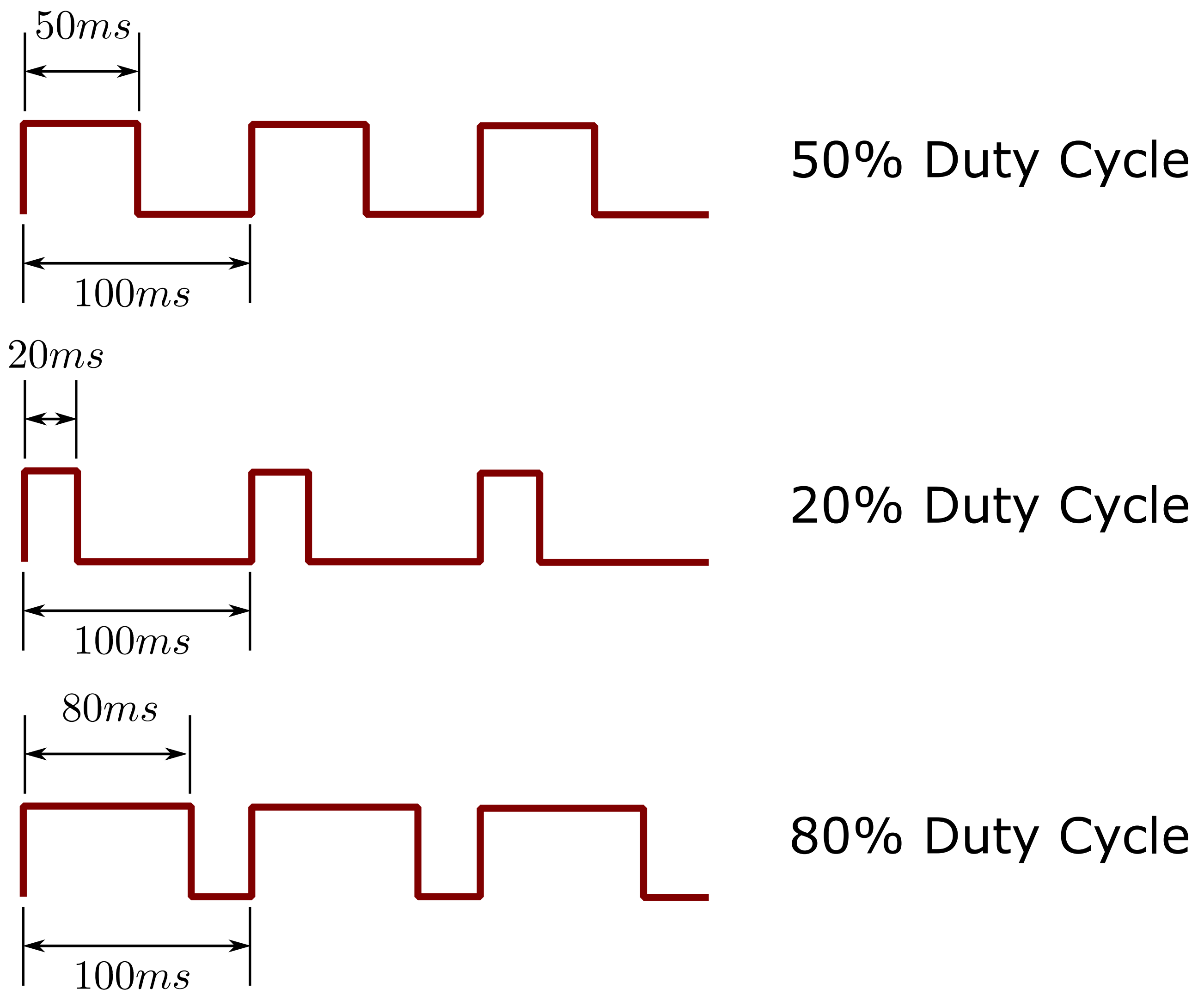
جامعة الكويت
KUWAIT UNIVERSITY

- Components of a PWM Signal
- Frequency is fixed for the application
- Pulse width is changed, hence the name <u>Pulse Width</u> Modulation
- Three quantities define PWM signal:
  - Pulse Width
  - Period (Frequency)
  - Voltage

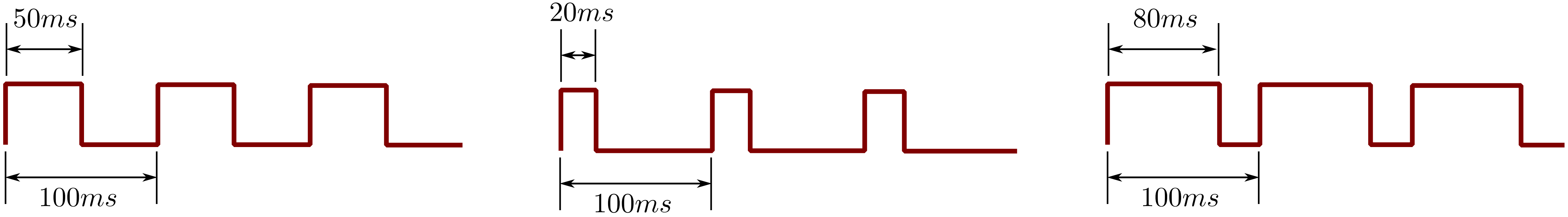
Duty Cyle = Pulse Width x Frequency x 100%



- Duty Cycle is the percentage of time the signal is HIGH



- PWM signal has a fixed frequency that is independent of the duty cycle
  - Configured to be a specific value based on the application



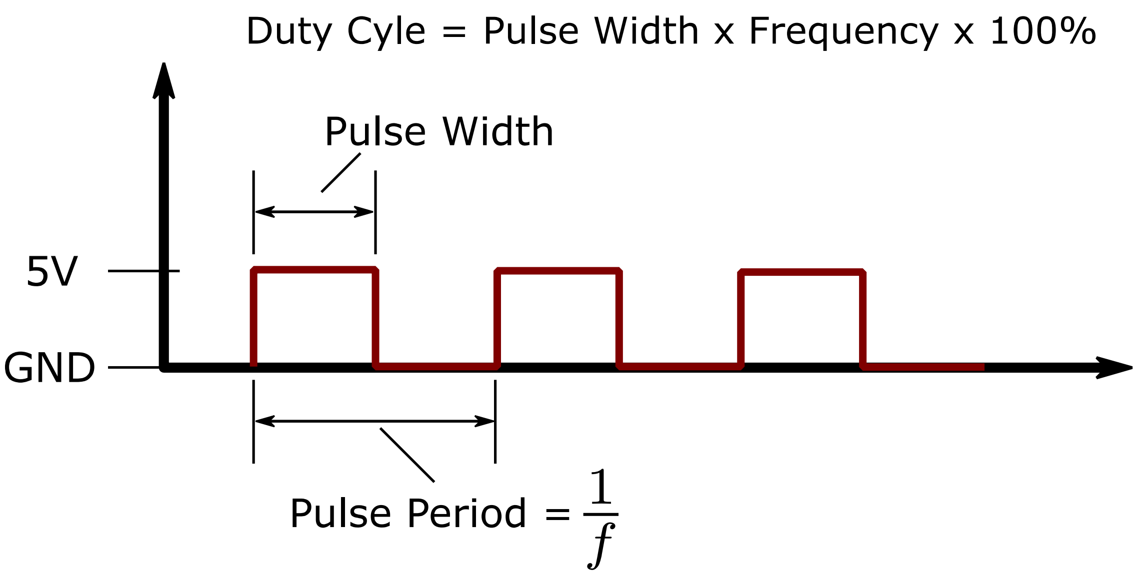
All these signals have a frequency of 1/0.1s = 10Hz

- PWM signal can be generated using a digital output pin by rapidly setting pin high/low (PWM with 50% Duty Cycle => square wave signal)

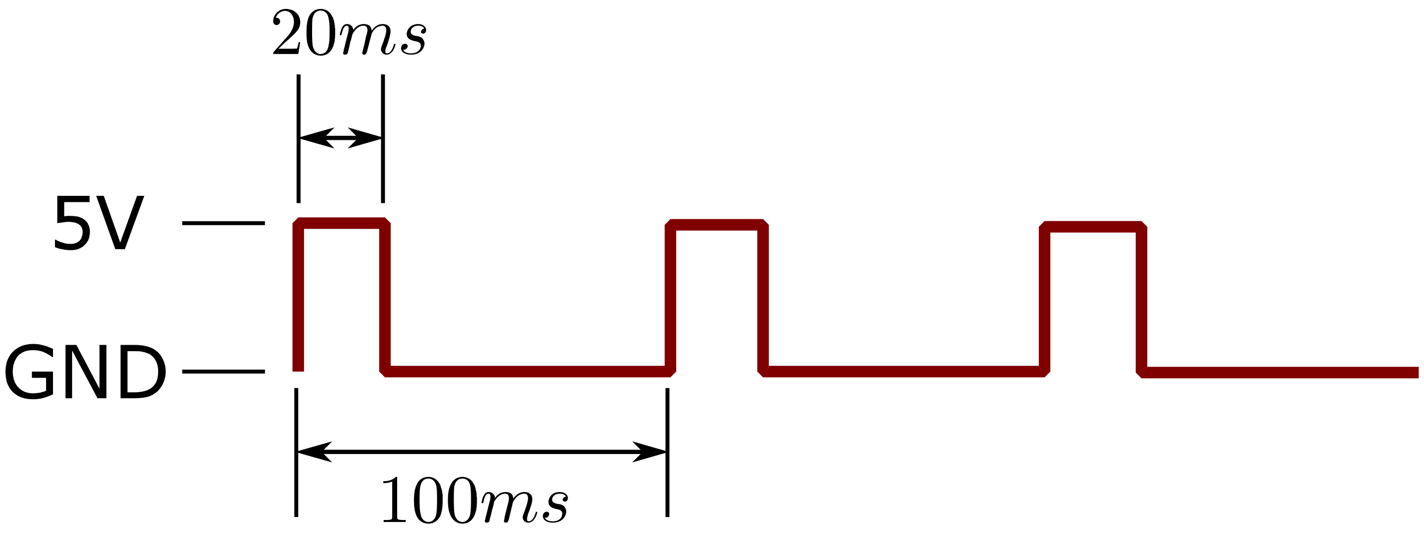
جامعة الكويت
KUWAIT UNIVERSITY

PWM Frequency

- Frequency must be high enough for AC component to be suppressed by the driven system
- For driving coils/windings, humming can occur for frequencies in the audible range. Aim for >25kHz
- Higher frequencies-> higher switching rates -> higher energy loss
- Avoid resonant frequencies of drive system



- What is the voltage of the PWM signal averaged over 1 cycle?

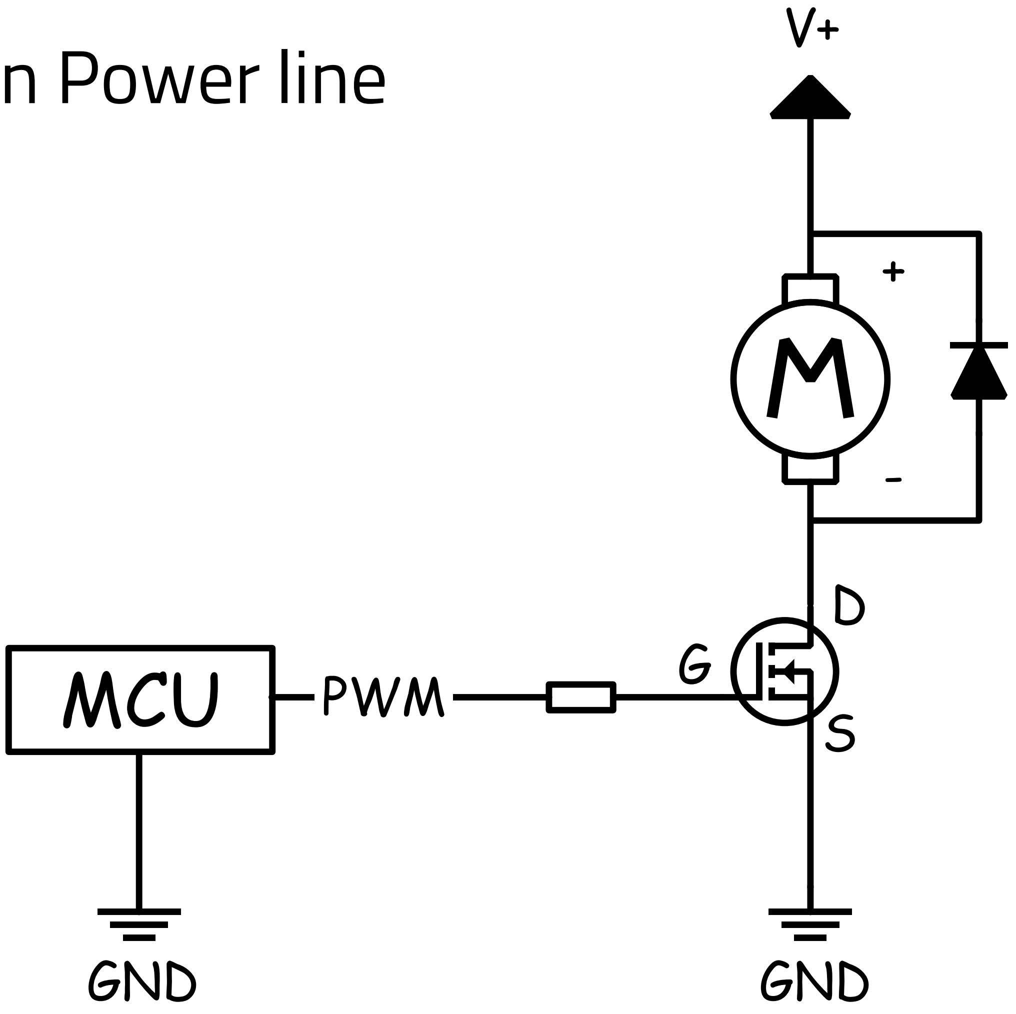


$$V_{AVG} = \frac{(5V)(20ms) + (0V)(80ms)}{100ms} = 1V$$

$$V_{AVG} = (Duty\ Cycle) \times V_{dd}$$

- Simple motor drive, one direction
- MOSFETs are able to switch much faster than PWM
- Signal PWM line replicated on Power line

$$V_{AVG} = (Duty\,Cycle) \times V_+$$



- PWM using Timer Module
  - Simple Implementation
  - Using Output Compare Principle
  - 32-bit or 16-bit timer
- PWM Using PWM Module
  - Provides PWM on all timers/channels
  - On Advanced Control Timer 1
    - Combined in single action or complementary pairs
    - Provide Dead-band delays (prevents shoot through; shorting)
    - Timer synchronization of PWM blocks
    - 16-bit timer

جامعة الكويت
KUWAIT UNIVERSITY

- The STM32F401RE has up to 11 timers

Table 4. Timer feature comparison

Timer type	Timer	Counter resolution	Counter type	Prescaler factor	DMA request generation	Capture/ compare channels	Complementary output	Max. interface clock (MHz)	Max. timer clock (MHz)
Advanced-control	TIM1	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	Yes	84	84
General purpose	TIM2, TIM5	32-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM3, TIM4	16-bit	Up, Down, Up/down	Any integer between 1 and 65536	Yes	4	No	42	84
	TIM9	16-bit	Up	Any integer between 1 and 65536	No	2	No	84	84
	TIM10, TIM11	16-bit	Up	Any integer between 1 and 65536	No	1	No	84	84

Timer Channel

- Timers interface with pins through channels
- On STM32F4x, a timer has up to four channels
- There is a specific pin associated with each channel
- A pin might be associated to more than one timer
- For Example: Look at PA2 and PA3

Table 9. Alternate function mapping

Port	AF00	AF01	AF02	AF03	AF04	AF05	AF06	AF07	AF08	AF09	AF10	AF11	AF12	AF13	AF14	AF15
	SYS_AF	TIM1/TIM2	TIM3/ TIM4/ TIM5	TIM9/ TIM10/ TIM11	I2C1/I2C2/ I2C3	SPI1/SPI2/ I2S2/SPI3/ I2S3/SPI4	SPI2/I2S2/ SPI3/ I2S3	SPI3/I2S3/ USART1/ USART2	USART6	I2C2/ I2C3	OTG1_FS		SDIO			
PA0	-	TIM2_CH1/ TIM2_ETR	TIM5_CH1	-	-	-	-	USART2_ CTS	-	-	-	-	-	-	-	EVENT OUT
PA1	-	TIM2_CH2	TIM5_CH2	-	-	-	-	USART2_ RTS	-	-	-	-	-	-	-	EVENT OUT
PA2	-	TIM2_CH3	TIM5_CH3	TIM9_CH1	-	-	-	USART2_ TX	-	-	-	-	-	-	-	EVENT OUT
PA3	-	TIM2_CH4	TIM5_CH4	TIM9_CH2	-		-	USART2_ RX	-	-	-	-	-	-	-	EVENT OUT

جامعة الكويت
KUWAIT UNIVERSITY