Carnegie Mellon University Qatar

15-348: Embedded Systems Lecture 11

Fall 2022

Ryan Riley
(based on original slides by Saquib Razak)

1

1

Where are we?

- Recently
  - Input Capture
- Today
  - More About the timers (input capture is just one thing they can do)
- Reading
  - Pages 707 – 715 in the launchpad datasheet
  - Pages 305 – 310 in the book

2

2

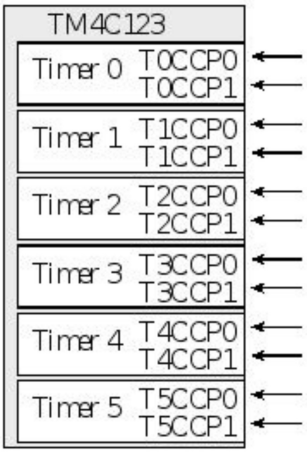
Fixing Some Mis-statements from Last Class

So many things I don't know

3

3

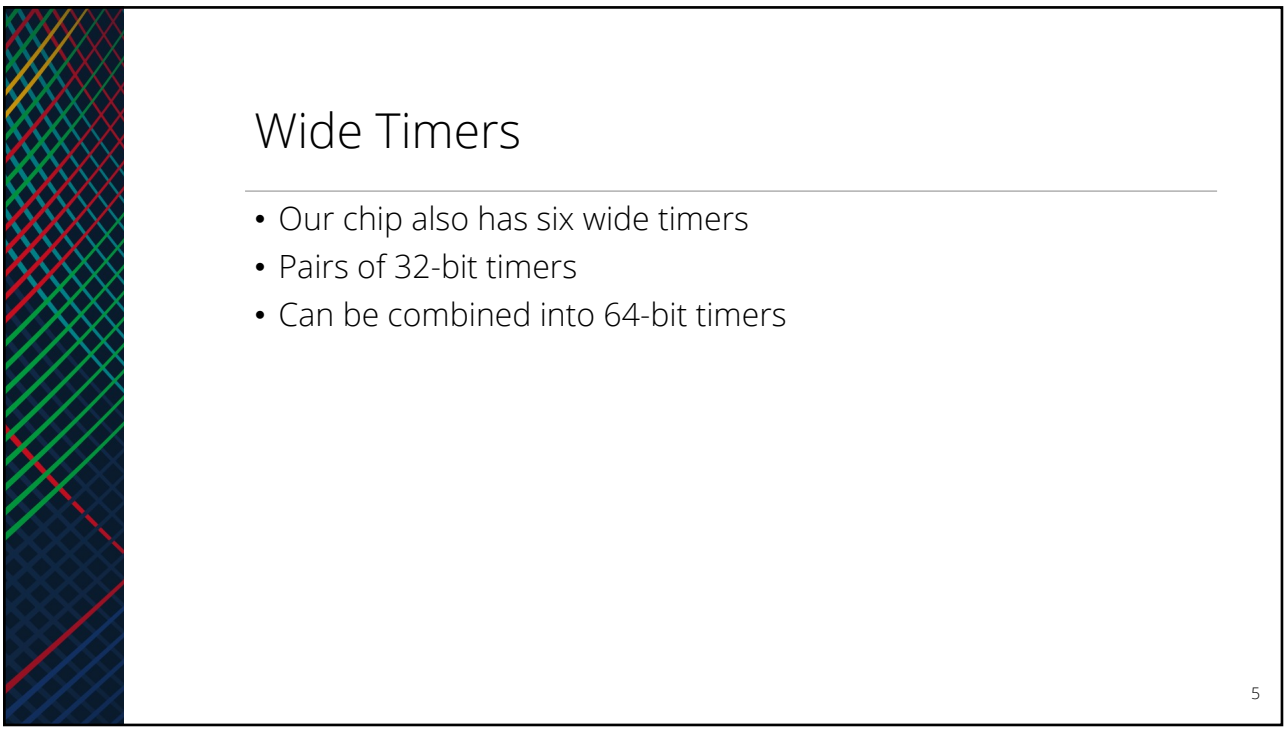
TimerA and TimerB



- Each timer is actually *two*, 16-bit timers
  - Timer 0A
  - Timer 0B
- Configured using the same control register, so they can be kept in perfect sync
- Can be combined to make a 32-bit timer

4

4

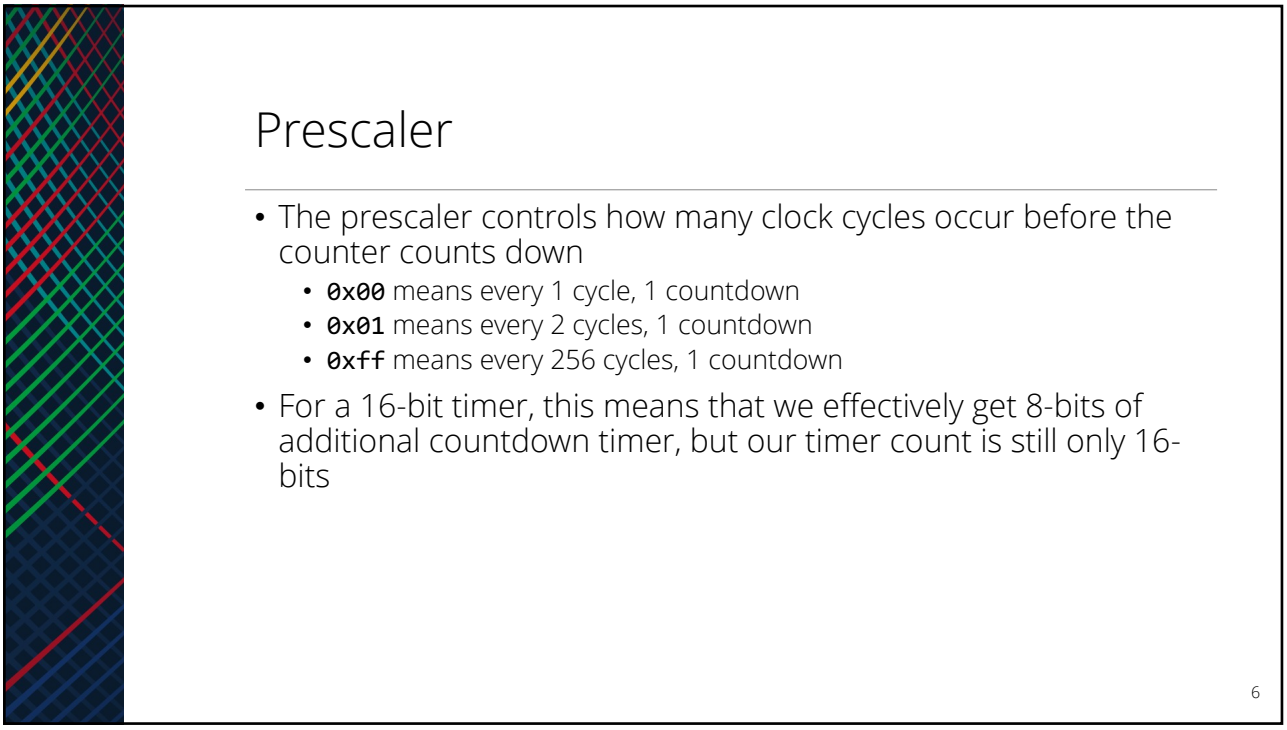


Wide Timers

- Our chip also has six wide timers
- Pairs of 32-bit timers
- Can be combined into 64-bit timers

5

5



Prescaler

- The prescaler controls how many clock cycles occur before the counter counts down
  - `0x00` means every 1 cycle, 1 countdown
  - `0x01` means every 2 cycles, 1 countdown
  - `0xff` means every 256 cycles, 1 countdown
- For a 16-bit timer, this means that we effectively get 8-bits of additional countdown timer, but our timer count is still only 16-bits

6

6

General Purpose Timer Module (GPTM) Overview

Timers in our Chip

- Six 16/32 bit modules
- Six 32/64 bit modules
- Each module has:
  - Two free running counters – TimerA and TimerB
  - Can be used individually (12, 16bit timers and 12, 32bit timers)
  - TimerA and TimerB can be combined to produce six 32bit counter and six 64bit counters
  - Or any combination
- Each 16-bit timer has an 8-bit prescalar
- Each 32-bit timer has a 16-bit prescalar

Timer Pin Connections

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
T1CCP0	30 58	PF2 (7) PB4 (7)	I/O	TTL	16/32-Bit Timer 1 Capture/Compare/PWM 0.
T1CCP1	31 57	PF3 (7) PB5 (7)	I/O	TTL	16/32-Bit Timer 1 Capture/Compare/PWM 1.
T2CCP0	5 45	PF4 (7) PB0 (7)	I/O	TTL	16/32-Bit Timer 2 Capture/Compare/PWM 0.
T2CCP1	46	PB1 (7)	I/O	TTL	16/32-Bit Timer 2 Capture/Compare/PWM 1.
T3CCP0	47	PB2 (7)	I/O	TTL	16/32-Bit Timer 3 Capture/Compare/PWM 0.
T3CCP1	48	PB3 (7)	I/O	TTL	16/32-Bit Timer 3 Capture/Compare/PWM 1.
T4CCP0	52	PC0 (7)	I/O	TTL	16/32-Bit Timer 4 Capture/Compare/PWM 0.
T4CCP1	51	PC1 (7)	I/O	TTL	16/32-Bit Timer 4 Capture/Compare/PWM 1.
T5CCP0	50	PC2 (7)	I/O	TTL	16/32-Bit Timer 5 Capture/Compare/PWM 0.
T5CCP1	49	PC3 (7)	I/O	TTL	16/32-Bit Timer 5 Capture/Compare/PWM 1.
WT0CCP0	16	PC4 (7)	I/O	TTL	32/64-Bit Wide Timer 0 Capture/Compare/PWM 0.
WT0CCP1	15	PC5 (7)	I/O	TTL	32/64-Bit Wide Timer 0 Capture/Compare/PWM 1.
WT1CCP0	14	PC6 (7)	I/O	TTL	32/64-Bit Wide Timer 1 Capture/Compare/PWM 0.
WT1CCP1	13	PC7 (7)	I/O	TTL	32/64-Bit Wide Timer 1 Capture/Compare/PWM 1.
WT2CCP0	61	PD0 (7)	I/O	TTL	32/64-Bit Wide Timer 2 Capture/Compare/PWM 0.
WT2CCP1	62	PD1 (7)	I/O	TTL	32/64-Bit Wide Timer 2 Capture/Compare/PWM 1.
WT3CCP0	63	PD2 (7)	I/O	TTL	32/64-Bit Wide Timer 3 Capture/Compare/PWM 0.
WT3CCP1	64	PD3 (7)	I/O	TTL	32/64-Bit Wide Timer 3 Capture/Compare/PWM 1.
WT4CCP0	43	PD4 (7)	I/O	TTL	32/64-Bit Wide Timer 4 Capture/Compare/PWM 0.
WT4CCP1	44	PD5 (7)	I/O	TTL	32/64-Bit Wide Timer 4 Capture/Compare/PWM 1.
WT5CCP0	53	PD6 (7)	I/O	TTL	32/64-Bit Wide Timer 5 Capture/Compare/PWM 0.
WT5CCP1	10	PD7 (7)	I/O	TTL	32/64-Bit Wide Timer 5 Capture/Compare/PWM 1.
T0CCP0	1 28	PB6 (7) PF0 (7)	I/O	TTL	16/32-Bit Timer 0 Capture/Compare/PWM 0.
T0CCP1	4 29	PB7 (7) PF1 (7)	I/O	TTL	16/32-Bit Timer 0 Capture/Compare/PWM 1.

What Can You Do With Them?

- Input capture
  - Measure input signals – Period, Frequency
- Free-running timer
  - Measure time passed
  - Trigger interrupts
- Output compare
  - Produce timed output pulses

Input Capture

We already did this. So this section is empty.

11

11

Free-Running Timer

Run Timer0A, Run!

12

12

Countdown Mode

- Very similar to SysTick
- Configure a interval load value (GPTMTnILR) and countdown to 0
- Can trigger an interrupt

- One-shot mode: Countdown to 0 once, then don't do it again
- Periodic mode: Countdown to 0, then reload, and keep doing it

- If using a prescaler, it controls how many cycles it takes for one countdown to occur

13

13

Count-up Mode

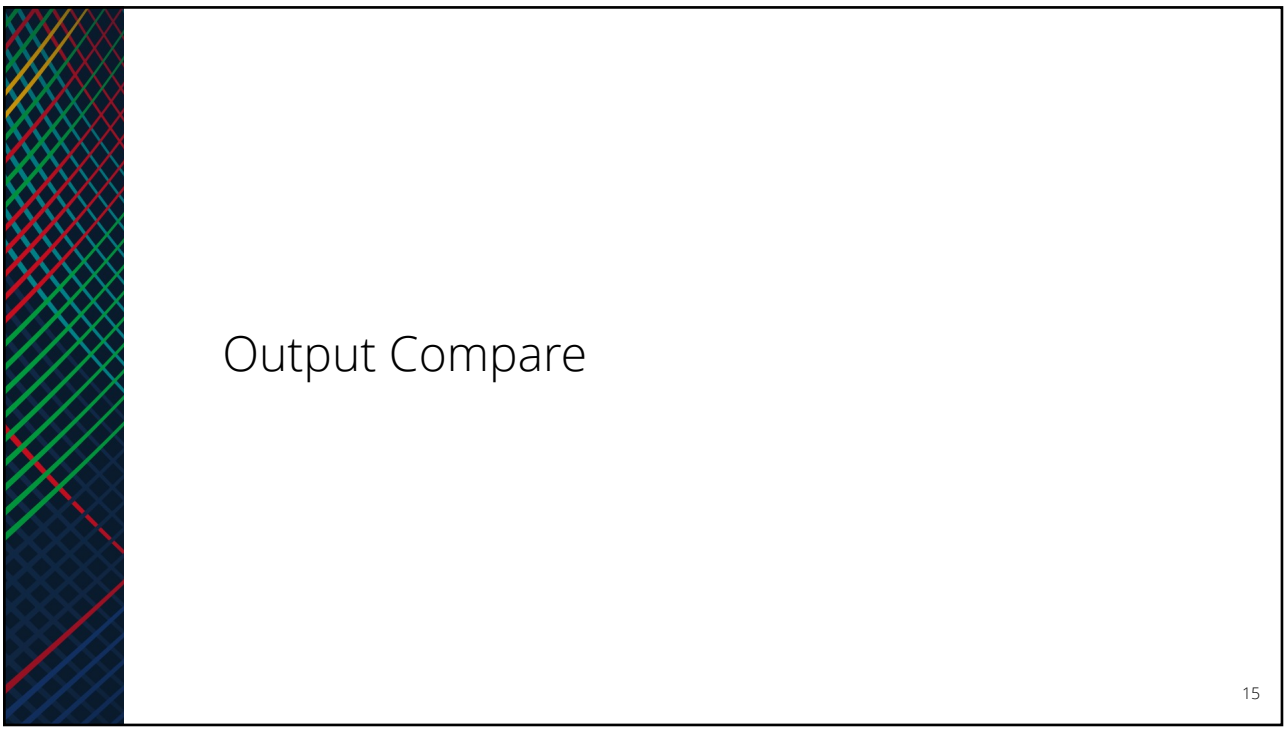
- Start at 0, count up to interval load value (GPTMTnILR)
- Can trigger an interrupt

- One-shot mode: Count-up up to interval load value once, then don't do it again
- Periodic mode: Count-up to interval load value, then reset to 0 and keep doing it

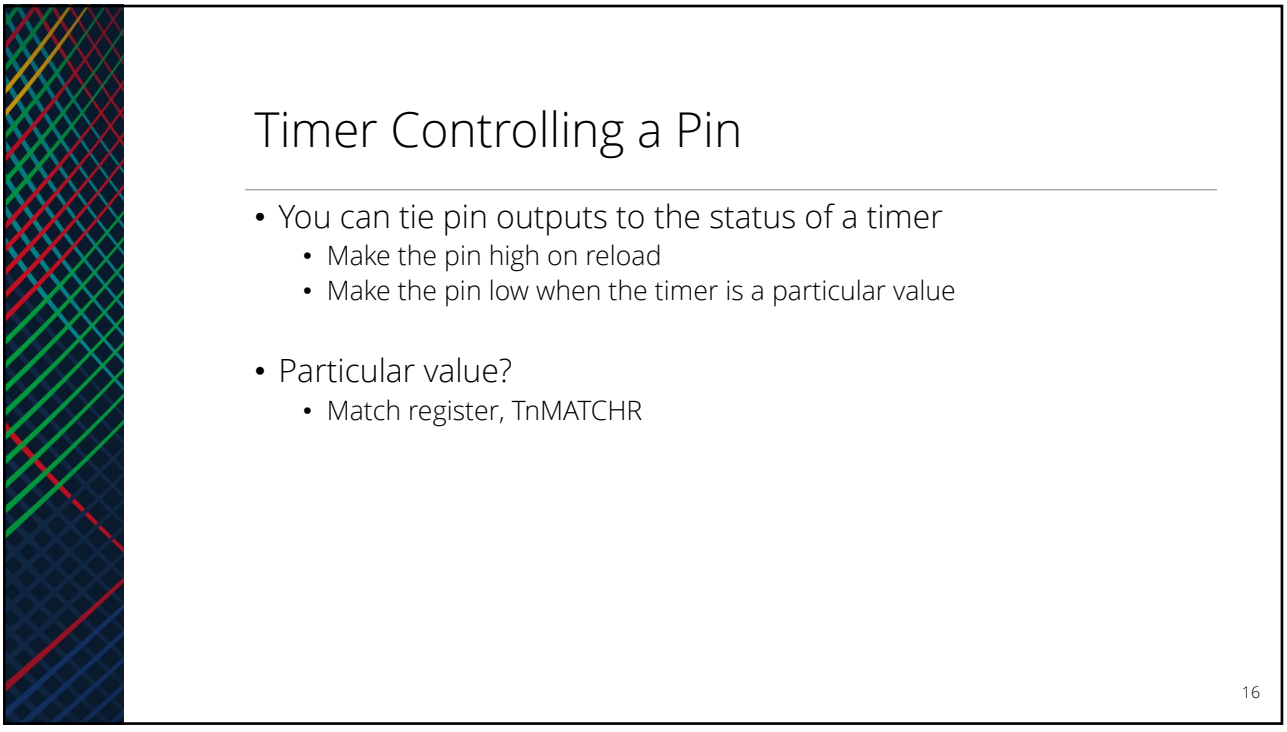
- If using a prescaler, it serves as the upper bits of the timer (bits 17-24 in 24-bit mode, for example)

14

14

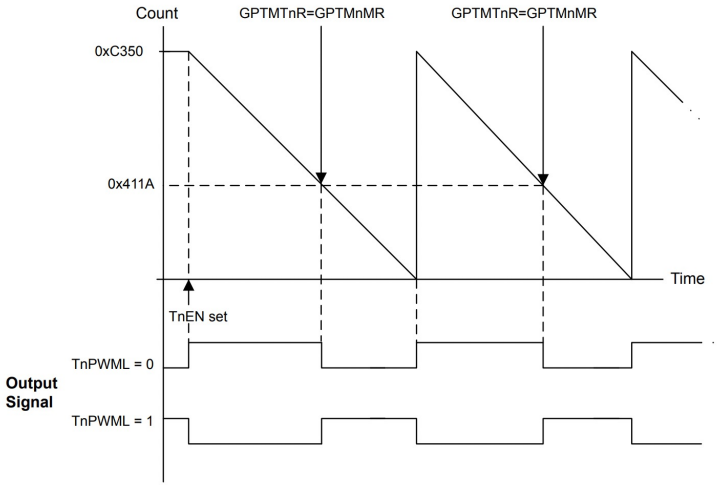

Output Compare
15

15


Timer Controlling a Pin
• You can tie pin outputs to the status of a timer
• Make the pin high on reload
• Make the pin low when the timer is a particular value
• Particular value?
• Match register, TnMATCHR
16

16

Example of Timer Controlling a Pin



17

17

Why Do This?

- When you need an output signal with a specific frequency and duty-cycle
- Pulse Width Modulation (PWM)
- Example: Servo motor
- Nice book example in Section 6.3.1
- There are also dedicated modules for this: Section 6.3.2

18

18

Example

19

Warning: Incoming Excercise

- Get the data sheet open to page 726
- Get in groups of 2 or 3

What does the following code configure the timer to do?

20

Code Tracing

```
void setup_timer0()
{
        SYSCTL_RCGCTIMER_R |= 0x01;
        TIMER0_CTL_R &= ~0x00000001;
        TIMER0_CFG_R = 0x00000000;
        TIMER0_TAMR_R = 0x00000002;
        TIMER0_TAILR_R = 0x4C4B400;
        TIMER0_TAPR_R = 0;
        TIMER0_ICR_R = 0x00000001;
        TIMER0_IMR_R |= 0x00000001;
        NVIC_PRI4_R = (NVIC_PRI4_R & 0x00FFFFFF) | 0x80000000;
        NVIC_EN0_R = 1 << 19;
        TIMER0_CTL_R |= 0x00000001;
}
```

21

21

Code Tracing (Solution)

```
void setup_timer0()
{
        SYSCTL_RCGCTIMER_R |= 0x01;
        TIMER0_CTL_R &= ~0x00000001;
        TIMER0_CFG_R = 0x00000000;
        TIMER0_TAMR_R = 0x00000002;
        TIMER0_TAILR_R = 0x4C4B400;
        TIMER0_TAPR_R = 0;
        TIMER0_ICR_R = 0x00000001;
        TIMER0_IMR_R |= 0x00000001;
        NVIC_PRI4_R = (NVIC_PRI4 R & 0x00FFFFFF) | 0x80000000;
        NVIC_EN0_R = 1 << 19;
        TIMER0_CTL_R |= 0x00000001;
}
```

22

22

Code Tracing (Solution)

```
void setup_timer0()
{
        SYSCTL_RCGCTIMER_R |= 0x01; // Activate Timer 0
        TIMER0_CTL_R &= ~0x00000001; // Disable timer 0 during configuration
        TIMER0_CFG_R = 0x00000000; // Configure for 32-bit Timer
        TIMER0_TAMR_R = 0x00000002; // Configure for periodic mode
        TIMER0_TAILR_R = 0x4C4B400; // Start value for count down (1 second)
        TIMER0_TAPR_R = 0; // No prescale
        TIMER0_ICR_R = 0x00000001; // Clear Timer0A timeout flag
        TIMER0_IMR_R |= 0x00000001; // Unmask the Timer0 interrupt
        NVIC_PRI4_R = (NVIC_PRI4_R & 0x00FFFFFF) | 0x80000000; // Prio 4
        NVIC_EN0_R = 1 << 19; // Enable timer0 interrupt
        TIMER0_CTL_R |= 0x00000001; // Enable Timer 0
}
```