COMP-462
Embedded Systems

Lecture 6: SysTick Timer

Agenda

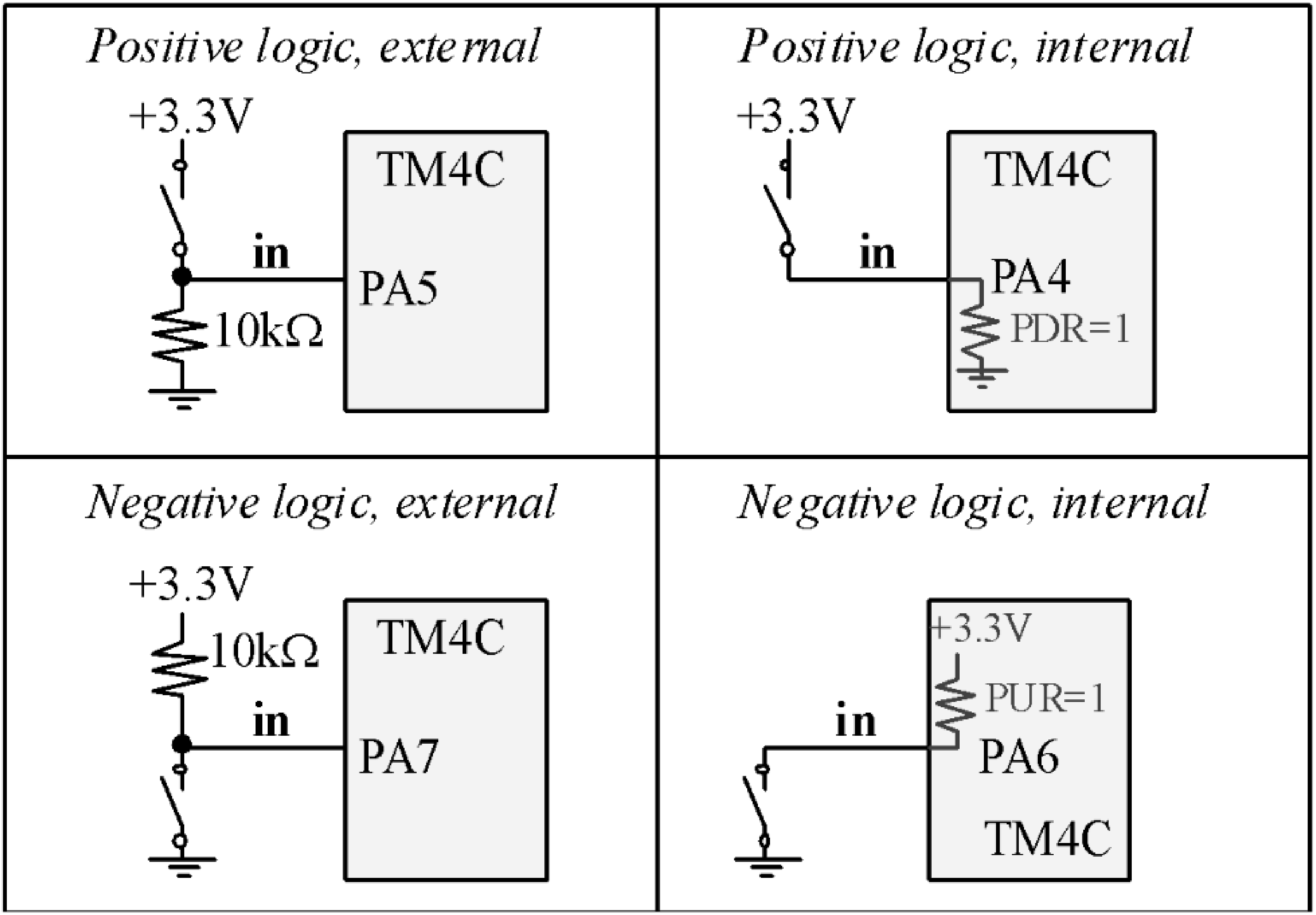
❑ Recap

- ❖ Subroutines and Parameter Passing
  - o AAPCS Convention
- ❖ Indexed Addressing and Pointers
  - o In C: Address of (&), Pointer to (*)
- ❖ Data Structures: Arrays, Strings
  - o Length: hardcoded vs. embedded vs. sentinel
  - o Array access: indexed vs. pointer arithmetic
- ❖ Functional Debugging

❑ Outline

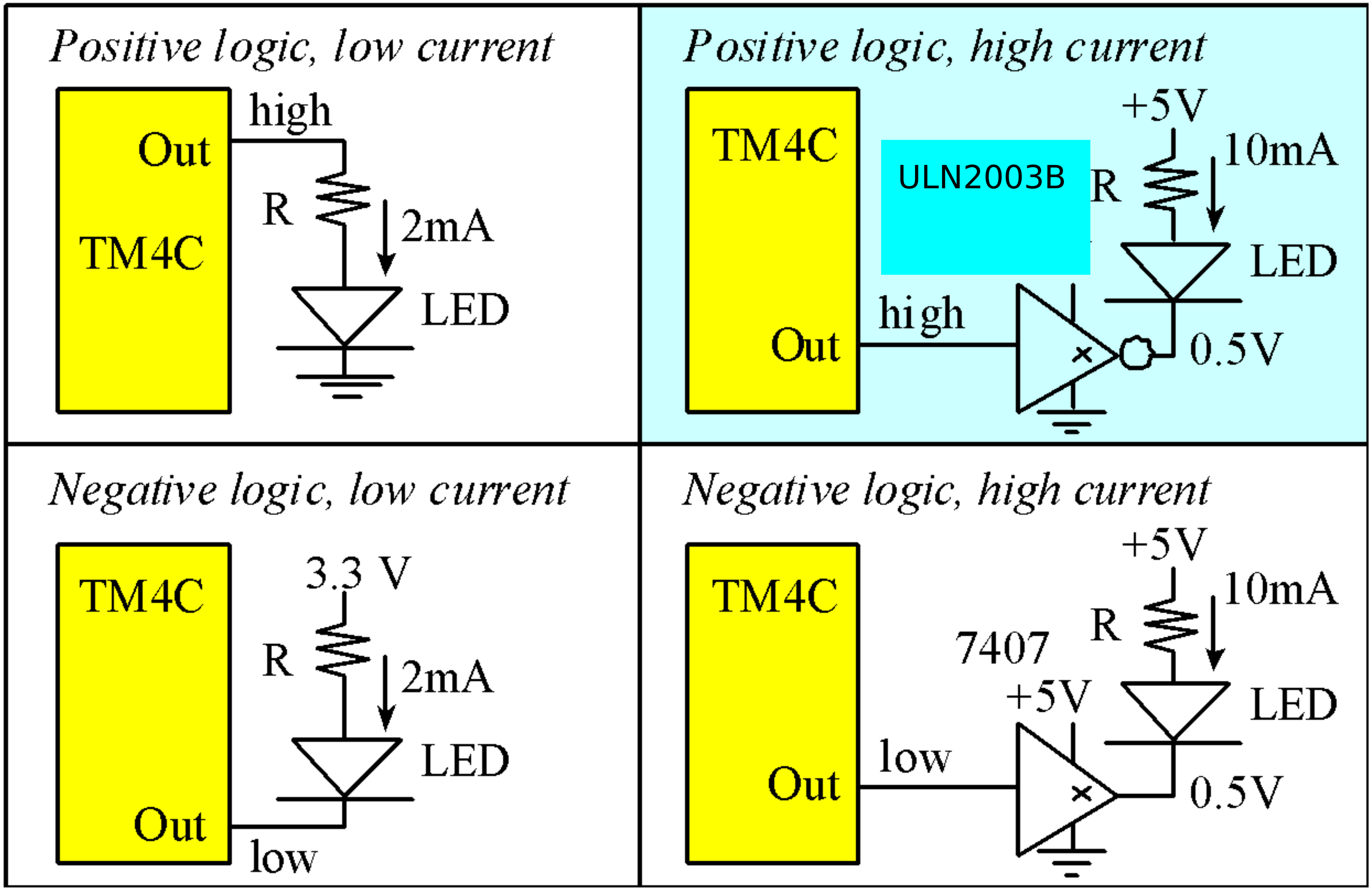
- ❖ SysTick Timer

Switch Interface



LED interfaces

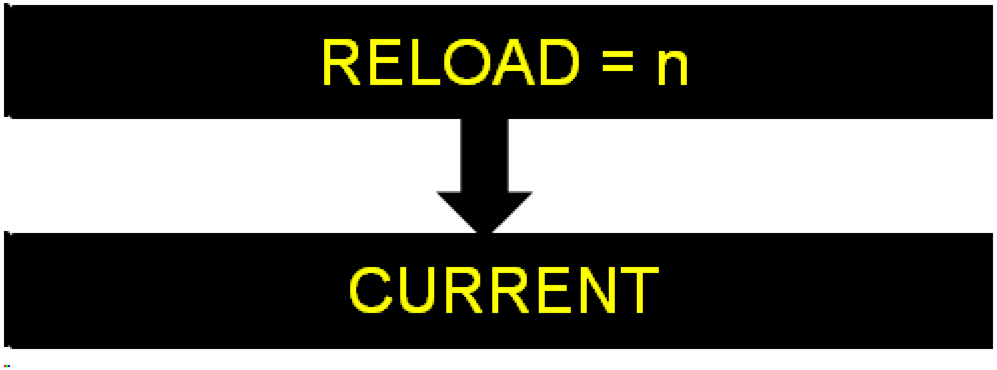
Replace ULN2003 wherever you see 7406 (they work the same)


Positive logic, low current
Out
TM4C
high
R
2mA
LED
Positive logic, high current
TM4C
Out
ULN2003B
high
R
+5V
10mA
LED
0.5V
Negative logic, low current
TM4C
Out
3.3 V
R
2mA
LED
low
Negative logic, high current
TM4C
Out
low
7407
+5V
+5V
R
10mA
LED
0.5V

Know voltage, current, power, Ohm's Law

SysTick Timer (new stuff)

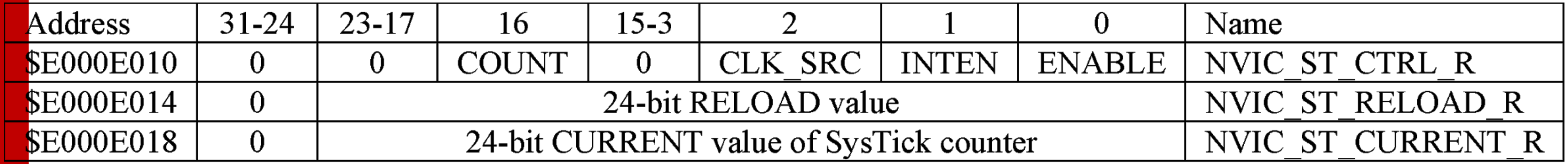
- Timer/Counter operation
  - 24-bit counter *decrements* at bus clock frequency
    - With 80 MHz bus clock, decrements every 12.5 ns
  - Counting is from *n* → 0
    - Setting *n* appropriately will make the counter a modulo *n*+1 counter. That is:
      - **`next_value = (current_value-1) mod (n+1)`**
      - **`Sequence: n,n-1,n-2,n-3… 2,1,0,n,n-1…`**



RELOAD = n

CURRENT

SysTick Timer



Address	31-24	23-17	16	15-3	2	1	0	Name
$E000E010	0	0	COUNT	0	CLK_SRC	INTEN	ENABLE	NVIC_ST_CTRL_R
$E000E014	0	24-bit RELOAD value						NVIC_ST_RELOAD_R
$E000E018	0	24-bit CURRENT value of SysTick counter						NVIC_ST_CURRENT_R

- ❑ Initialization (4 steps)
  - ❖ Step1: Clear ENABLE to stop counter
  - ❖ Step2: Specify the RELOAD value
  - ❖ Step3: Clear counter by reading NVIC_ST_CURRENT_R
  - ❖ Step4: Set NVIC_ST_CTRL_R
    - o CLK_SRC = 1 (bus clock is the only option)
    - o INTEN = 0 for no interrupts
    - o ENABLE = 1 to enable

SysTick Timer

24-bit Countdown Timer

```
SysTick_Init
; disable SysTick during setup
    LDR R1, =NVIC_ST_CTRL_R
    MOV R0, #0                ; Clear Enable
    STR R0, [R1]
; set reload to maximum reload value
    LDR R1, =NVIC_ST_RELOAD_R
    LDR R0, =0x00FFFFFF;    ; Specify RELOAD value
    STR R0, [R1]            ; reload at maximum
; writing any value to CURRENT clears it
    LDR R1, =NVIC_ST_CURRENT_R
    MOV R0, #0
    STR R0, [R1]            ; clear counter
; enable SysTick with core clock
    LDR R1, =NVIC_ST_CTRL_R
    MOV R0, #0x0005    ; Enable but no interrupts (later)
    STR R0, [R1]       ; ENABLE and CLK_SRC bits set
    BX  LR
```

SysTick Timer

```
;------------SysTick_Wait------------
; Time delay using busy wait.
; Input: R0  delay parameter in units of the core clock
;        80 MHz(12.5 nsec each tick)
; Output: none
; Modifies: R1
SysTick_Wait
    SUB  R0, R0, #1   ; delay-1
    LDR  R1, =NVIC_ST_RELOAD_R
    STR  R0, [R1]     ; time to wait
    LDR  R1, =NVIC_ST_CURRENT_R
    STR  R0, [R1]     ; any value written to CURRENT clears
    LDR  R1, =NVIC_ST_CTRL_R
SysTick_Wait_loop
    LDR  R0, [R1]     ; read status
    ANDS R0, R0, #0x00010000 ; bit 16 is COUNT flag
    BEQ  SysTick_Wait_loop   ; repeat until flag set
    BX   LR
```

SysTick Timer

```
;------------SysTick_Wait10ms------------
; Call this routine to wait for R0*10 ms
; Time delay using busy wait. This assumes 80 MHz clock
; Input: R0  number of times to wait 10 ms before returning
; Output: none
; Modifies: R0
DELAY10MS  EQU 800000    ; clock cycles in 10 ms
SysTick_Wait10ms
    PUSH {R4, LR}               ; save R4 and LR
    MOVS R4, R0                 ; R4 = R0 = remainingWaits
    BEQ SysTick_Wait10ms_done   ; R4 == 0, done
SysTick_Wait10ms_loop
    LDR R0, =DELAY10MS          ; R0 = DELAY10MS
    BL  SysTick_Wait            ; wait 10 ms
    SUBS R4, R4, #1             ; remainingWaits--
    BHI SysTick_Wait10ms_loop   ; if(R4>0), wait another 10
ms
SysTick_Wait10ms_done
    POP {R4, PC}
```

SysTick Timer in C

```c
#define NVIC_ST_CTRL_R(*((volatile uint32_t *)0xE000E010))
#define NVIC_ST_RELOAD_R(*((volatile uint32_t *)0xE000E014))
#define NVIC_ST_CURRENT_R(*((volatile uint32_t *)0xE000E018))

void SysTick_Init(void){
 NVIC_ST_CTRL_R = 0; // 1) disable SysTick during setup
 NVIC_ST_RELOAD_R = 0x00FFFFFF; // 2) maximum reload value
 NVIC_ST_CURRENT_R = 0; // 3) any write to CURRENT clears it
 NVIC_ST_CTRL_R = 0x00000005; // 4) enable SysTick with core clock
}

// The delay parameter is in units of the 80 MHz core clock(12.5 ns)
void SysTick_Wait(uint32_t delay){
  NVIC_ST_RELOAD_R = delay-1;  // number of counts
  NVIC_ST_CURRENT_R = 0;       // any value written to CURRENT clears
  while((NVIC_ST_CTRL_R&0x00010000)==0){ // wait for flag
  }
}
// Call this routine to wait for delay*10ms
void SysTick_Wait10ms(uint32_t delay){
 unsigned long i;
 for(i=0; i<delay; i++){
   SysTick_Wait(800000); // wait 10ms
}
```