CPE 323
Intro to Embedded Computer Systems
Serial Communication (UART)

Aleksandar Milenkovic

milenka@uah.edu

Admin

oogle Drive | Quiz: Midterm Exam - Requires R | +

uah.instructure.com/courses/48289/quizzes/99219/take?preview=1

ture Notes | Cry... | CIS 371: Computer... | How to Refurbish Y...

Question 8 20 pts

Stack allocation, pointers, pointer arithmetic.

Consider the following C code snippet. Assume that the register SP at the beginning points to 0x1100. Answer the following questions. Assume all variables are allocated on the stack and in the order as they appear in the program (i, ia, mc, pli, pc). ASCII code for character '0' is 48 (0x30), for 'A' is 0x41, and for 'a' is 0x61.

```
1 int main( void ) {
2    volatile long int li[ 2 ] = {3, -4};
3    volatile int ia[ 2 ] = {-4, 5};
4    volatile char mc[ 4 ] = {'2', 'A', 'a', '3'};
5    volatile char *pc = mc;
6    volatile long int *pli = li+1;
7    pc = pc + 3;            //
8    *pc += *(pc-1) - 0x30;  //
9    *pli = *pli + 4;        //
10 }
```

33 + 61 − 30 = 0x64

Answer the following questions.

A. How many bytes is allocated on the stack for the variable declared in line 2? 8 bytes (enter decimal number)

B. How many bytes is allocated on the stack for the character array in line 4? 4 bytes (enter decimal number)

C. How many bytes is allocated on the stack for all variables in lines 2-6? 20 bytes (enter decimal number)

D. What is the value of mc[1] after initialization in line 4? (enter hex value starting with 0x)

E. What is the value of li? (enter hex value starting with 0x)

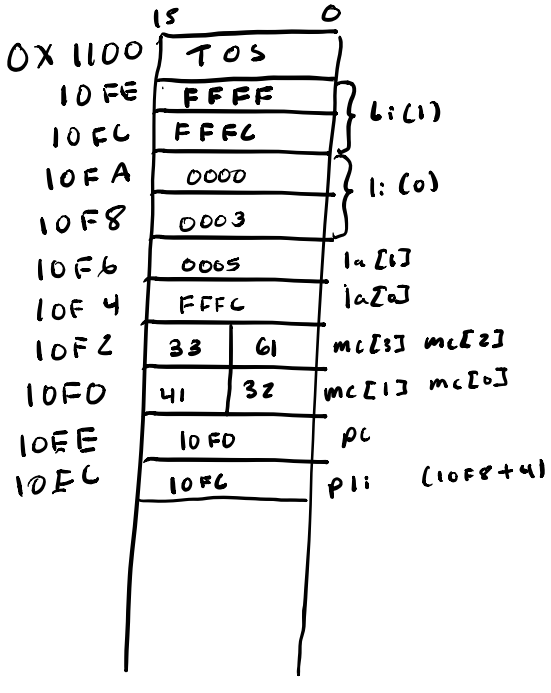
F. What is value of pc after the statement in line 5 is executed? 0X41 (enter hex value starting with 0x)

G. What is the value of pli after the statement in line 6 is executed? 10FC (enter hex value starting with 0x)

H. What is the value of pc after the statement in line 7 is executed? 10F3 (enter hex value starting with 0x)

I. What is the value of mc[3] after the statement in line 8 is executed? 0X64 (enter hex value starting with 0x)

J. What is the value of li[1] after the statement in line 9 is executed? (enter hex value starting with 0x)



S: .long

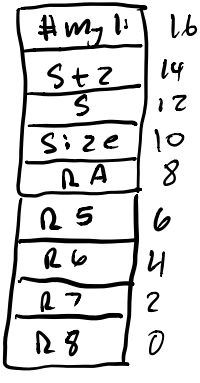
Question 9 20 pts

Design and write an MSP430 assembly language subroutine AddStoLINTA with the following prototype:

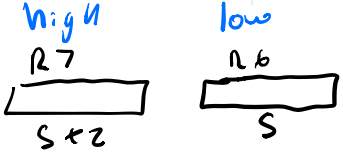
void AddStoLINTA(long int myli, long int S, unsigned int size)*

The subroutine adds a scalar *S* to each element of an array *myli* with *size* elements. Both the scalar *S* and the array *myli* are long signed integers (32-bit long). Assume that a caller program pushes the input parameters onto the stack, before calling the subroutine. The inputs are pushed onto the stack in the order they are listed in the function descriptor, i.e., the main program before calling the subroutine *AddStoLINTA* will execute the following sequence of instructions:

```
PUSH.W #myli  ;
PUSH.W  S+2   ;
PUSH.W  S     ;
PUSH.W  size  ;
```




```
AddStoLINTA:
    push R5
    push R6
    push R7
    push R8
    mov.w  16(SP), R5
    mov.w  10(SP), R8
    mov.w  @R5+, R6
    mov.w  @R5+, R7
    add.w  12(SP), R6
    addc.w 14(SP), R7
```



MSP430F5529 Block Diagram

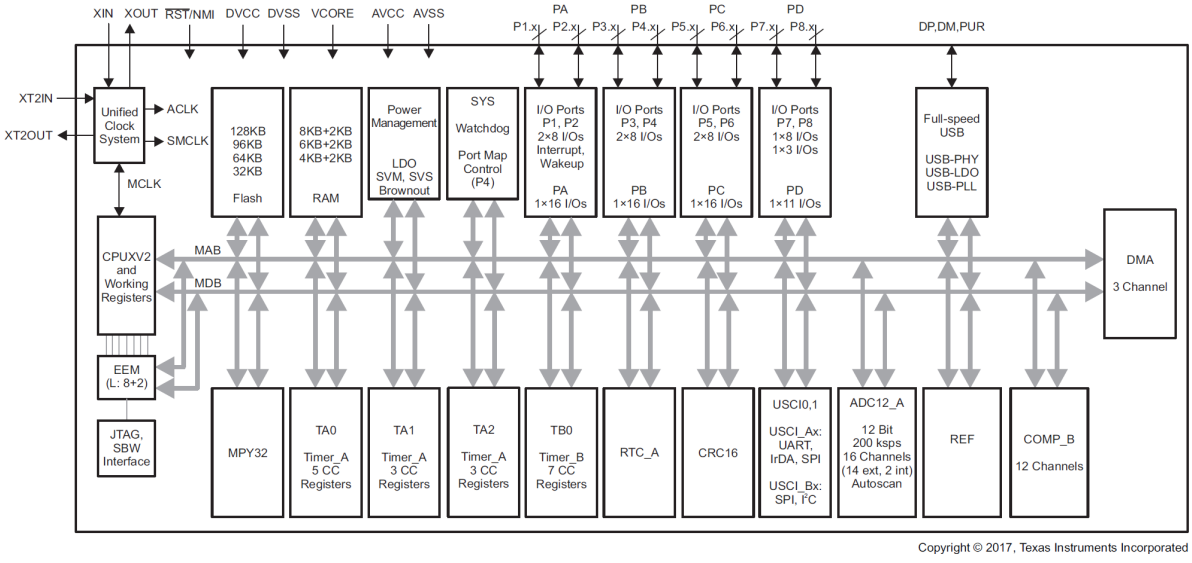


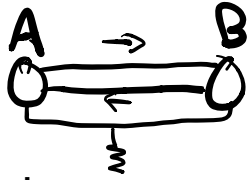
Figure 1-1. Functional Block Diagram – MSP430F5529IPN, MSP430F5527IPN, MSP430F5525IPN, MSP430F5521IPN

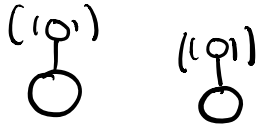
Communication

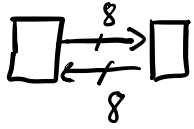
- Part of big 4
  - sense
  - process (compute)
  - store (memory)
  - communicate (UI, networks, ...)
- Communication in embedded systems
  - Between integrated circuits on PCB (e.g., I2C sensors)
  - Between development platform and a workstation
  - Between embedded systems

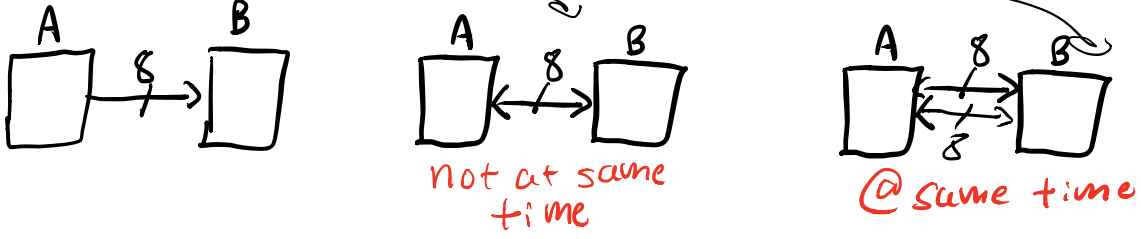
Types of Communication

- Wired vs. wireless
- Serial vs. parallel
- Synchronous vs. asynchronous
- Unidirectional (simplex) vs. bidirectional (half-duplex and full-duplex)







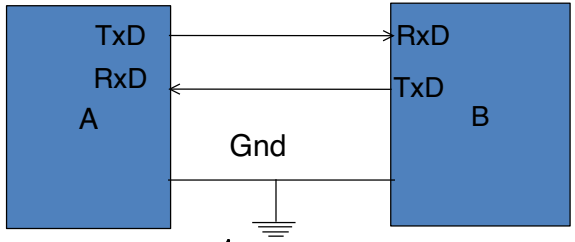


Serial Communication in MSP430

- Communication protocols
  - UART (Universal Asynchronous Receiver/Transmitter)
  - SPI (Serial Parallel Interface) – Synchronous, bidirectional
  - I^2C (Inter Integrated Circuit) – Synchronous, bidirectional, bus [SDA SCL]
  - Infrared
- Peripheral devices
  - USCI – Universal Serial Communication Interface 8-Bit
  - USI – Universal Serial Interface
  - USART – Universal Synchronous/Asynchronous Receiver/Transmitter

UART

TxD - Transmit data
RxD - Receive data

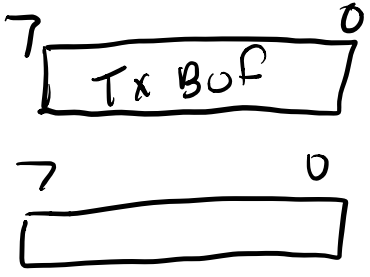


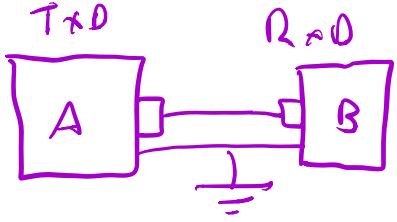
↳ have to have common ground.

Asynchronous Communication

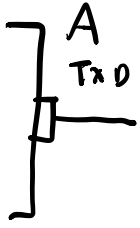
Character oriented

↳

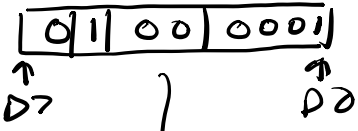


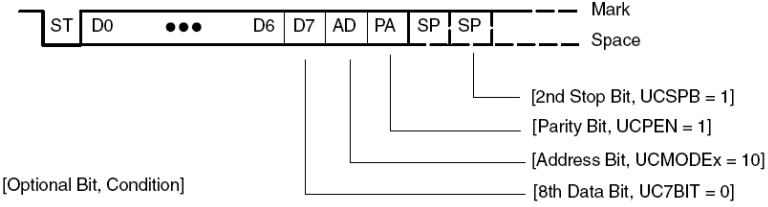


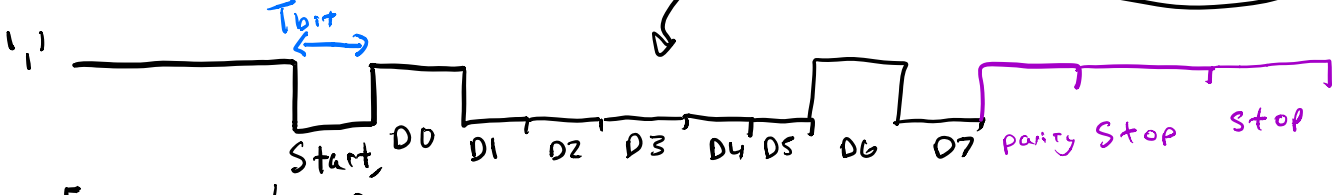
Character Format



Tx BUF = 'A' = 0x41







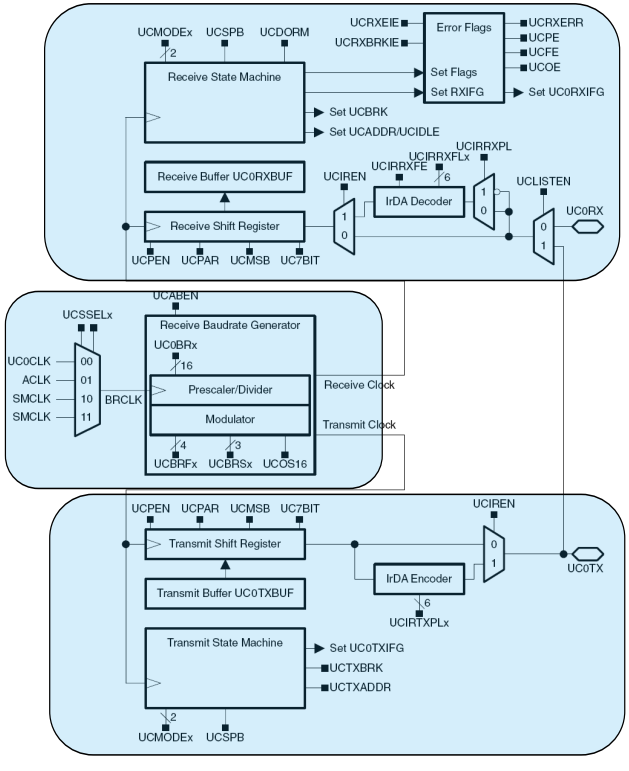
Even ↓ Parity bits: ↑ Odd

1 + 8 + 1 + 2 = 12 bits

Start, data, parity

12 · Tbit → time

Figure 19–1. USCI_Ax Block Diagram: UART Mode (UCSYNC = 0)



Baudrate 9,600 bps

14,400

19,200

38400

$$Tbit = \frac{1}{9600}$$

UC0TXBUF = 'A';

double buffering

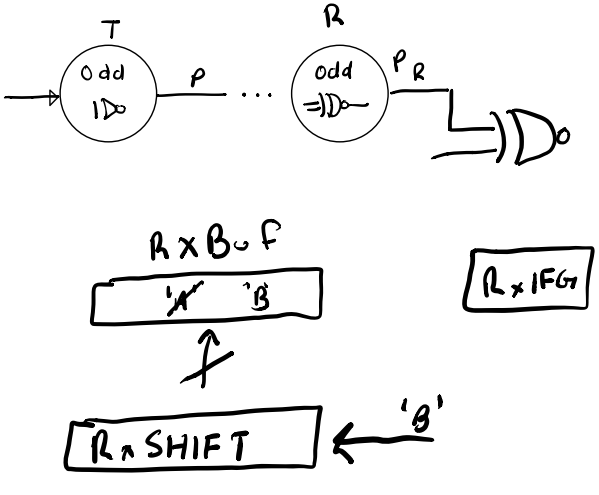
USCI_A0 Registers

Table 19–6. USCI_A0 Control and Status Registers

Register	Short Form	Register Type	Address	Initial State
USCI_A0 control register 0	UCA0CTL0	Read/write	060h	Reset with PUC
USCI_A0 control register 1	UCA0CTL1	Read/write	061h	001h with PUC
USCI_A0 Baud rate control register 0	UCA0BR0	Read/write	062h	Reset with PUC
USCI_A0 Baud rate control register 1	UCA0BR1	Read/write	063h	Reset with PUC
USCI_A0 modulation control register	UCA0MCTL	Read/write	064h	Reset with PUC
USCI_A0 status register	UCA0STAT	Read/write	065h	Reset with PUC
USCI_A0 Receive buffer register	UCA0RXBUF	Read	066h	Reset with PUC
USCI_A0 Transmit buffer register	UCA0TXBUF	Read/write	067h	Reset with PUC
USCI_A0 Auto Baud control register	UCA0ABCTL	Read/write	05Dh	Reset with PUC
USCI_A0 IrDA Transmit control register	UCA0IRTCTL	Read/write	05Eh	Reset with PUC
USCI_A0 IrDA Receive control register	UCA0IRRCTL	Read/write	05Fh	Reset with PUC
SFR interrupt enable register 2	IE2	Read/write	001h	Reset with PUC
SFR interrupt flag register 2	IFG2	Read/write	003h	00Ah with PUC

Transmit	Receive
Initialize USCI For UART communication F_{Baud} = 38,400 bps 8-bit data LSB First P Even } setting up registers Polling Read TxIFG IF TxBUF is empty (IFG=1) TxBUF = mychar;	Initialize USCI For UART Comm. F_{BaUD} = 38,400 bps 8-bit data LSB First P Even wait for RxIFG to be set; received char = RxBUF; ↑ reading the cleared

Error Conditions



Error Condition	Error Flag	Description
Framing error	UCFE	A framing error occurs when a low stop bit is detected. When two stop bits are used, both stop bits are checked for framing error. When a framing error is detected, the UCFE bit is set.
Parity error	UCPE	A parity error is a mismatch between the number of 1s in a character and the value of the parity bit. When an address bit is included in the character, it is included in the parity calculation. When a parity error is detected, the UCPE bit is set.
Receive overrun	UCOE	An overrun error occurs when a character is loaded into UCAxRXBUF before the prior character has been read. When an overrun occurs, the UCOE bit is set.
Break condition	UCBRK	When not using automatic baud rate detection, a break is detected when all data, parity, and stop bits are low. When a break condition is detected, the UCBRK bit is set. A break condition can also set the interrupt flag UCAxRXIFG if the break interrupt enable UCBRKIE bit is set.

Baud Rate Generation

- Definitions
  - BRCLK is input clock (ACLK, SMCLK, UCLK)
  - $F_{BITCLK}=F_{BAUD}$ is bit clocks (e.g., 38,400 bps) $T_{BITCLK} = 1/38,400$
  - $F_{BITCLK16}=16*F_{BAUD}$
- Oversampling mode (UCOS16=1)
  - BRCLK is divided to give BITCLK16, which is further divided by 16 to give BITCLK
- Low Frequency mode (UCOS16=0)
  - BRCLK is divided to give BITCLK

Baud Rate Generation

- Oversampling: $f_{baud} = 9600$ Hz, $f_{BRCLK} = 2^{20}$ Hz

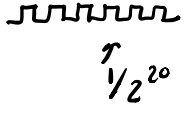
$f_{baud} = 16 \times 9{,}600 = 153.6$ kHz

$N = \frac{f_{BRCLK}}{f_{baud}} = 109.22$

$\frac{N}{16}$

$UCBRx = INT\left(\frac{N}{16}\right) = 6$

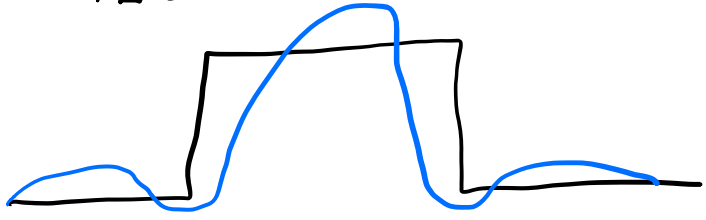
Oversampling



$N/16 = 6.83$

$T \approx 1/2^{20}$

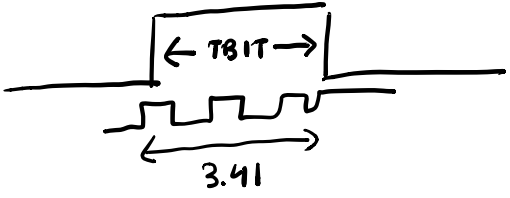
$UCBRFx = round\left(\left(\frac{N}{16} - INT\left(\frac{N}{16}\right)\right) \times 16\right) = 13$



Baud Rate Generation

- Low frequency is used when $f_{BRCLK} < 16 * f_{baud}$
- $f_{baud} = 9600$ Hz, $f_{BRCLK} = 2^{15}$ Hz (ACLK)

$$N = \frac{2^{15}}{9{,}600} = 3.41 < 16$$



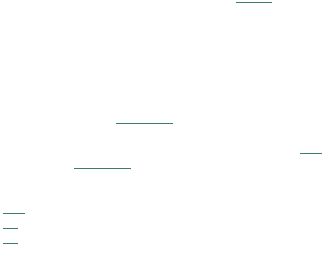
UCBRx = INT(N) = 3

UCBRSx = round((3.41 − 3) · 8) = 3

5 bits out of 8 will be $3 \times T_{BRCLK}$

3 bits out of 8 will be $4 \times T_{BRCLK}$

Echo a character using Polling



Echo a character using ISR

Display Real-Time Clock

Display Real-Time Clock (cont'd)