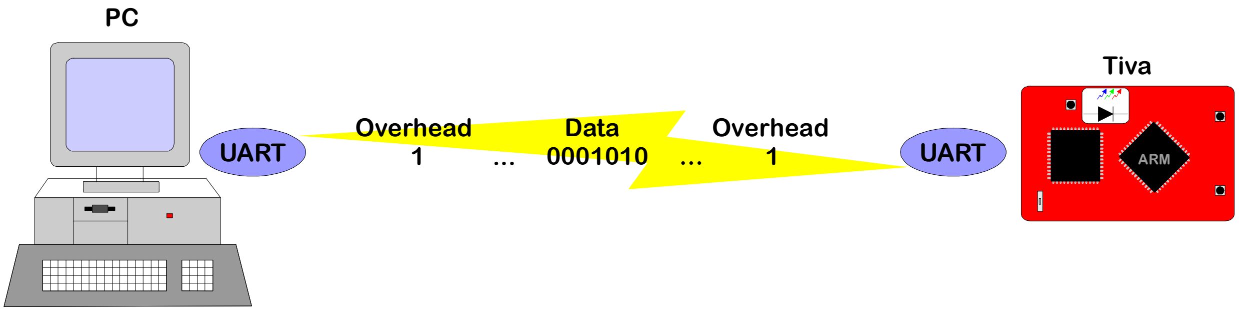
CSE 379

Serial Communications with the ARM Processor

Serial Communication

- Data is transferred one bit at a time
- Encapsulation
  - Data is framed between control bits
- UART
  - Universal Asynchronous Receiver/Transmitter
  - Hardware responsible for serial communication



UART

- There are seven UARTs on the ARM board
  - We'll use UART0

UART Use

- Procedure
  - Initialize Serial Port
  - Read/Transmit Data

UART Initialization

- We will provide you with initialization code in C
- Use this to get Lab #3 up and running
- After it works, write your own initialization routine in assembly

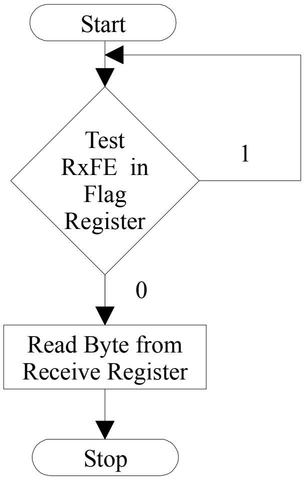
- ## C Code

```c
void serial_init(void)
{
        /* When translating the following to assembly   */
        /* it is advised to use LDR and STR as opposed  */
        /* to LDRB and STRB.                            */
        /* Provide clock to UART0  */
        (*((volatile uint32_t *)(0x400FE618))) = 1;
        /* Enable clock to PortA  */
        (*((volatile uint32_t *)(0x400FE608))) = 1;
        /* Disable UART0 Control  */
        (*((volatile uint32_t *)(0x4000C030))) = 0;
        /* Set UART0_IBRD_R for 115,200 baud */
        (*((volatile uint32_t *)(0x4000C024))) = 8;
        /* Set UART0_FBRD_R for 115,200 baud */
        (*((volatile uint32_t *)(0x4000C028))) = 44;
        /* Use System Clock */
        (*((volatile uint32_t *)(0x4000CFC8))) = 0;
        /* Use 8-bit word length, 1 stop bit, no parity */
        (*((volatile uint32_t *)(0x4000C02C))) = 0x60;
        /* Enable UART0 Control  */
        (*((volatile uint32_t *)(0x4000C030))) = 0x301;

        /* The OR operation sets the bits that are OR'ed */
        /* with a 1.  To translate the following lines   */
        /* to assembly, load the data, OR the data with  */
        /* the mask and store the result back.           */

        /* Make PA0 and PA1 as Digital Ports  */
        (*((volatile uint32_t *)(0x4000451C))) |= 0x03;
        /* Change PA0,PA1 to Use an Alternate Function  */
        (*((volatile uint32_t *)(0x40004420))) |= 0x03;
        /* Configure PA0 and PA1 for UART  */
        (*((volatile uint32_t *)(0x4000452C))) |= 0x11;
}
```

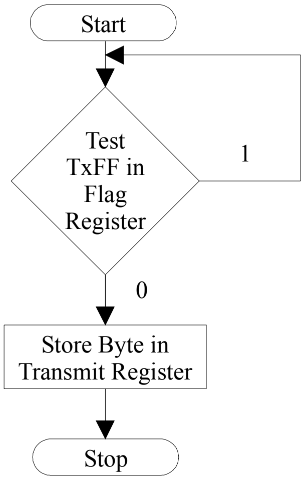
- ## Details
  - ☞ Initialization Notes
    - ↳ For further details refer to the *Tiva™ TM4C123GH6PM Microcontroller Data Sheet – Chapter 14*

UART Use

- Receiving Data

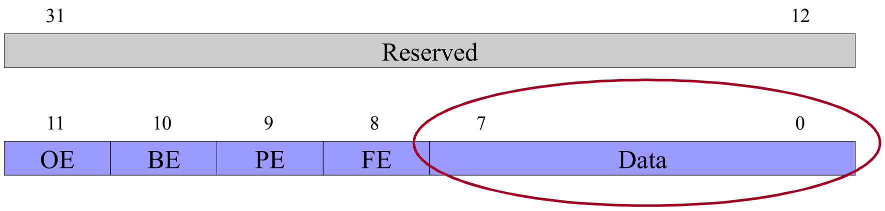


- Transmitting Data



UART Data Register

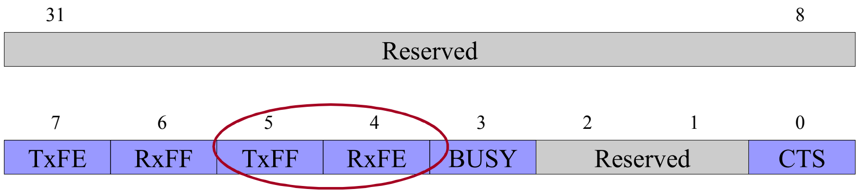
- UART0
  - UARTDR
    - Address: 0x4000C000



- Details
  - Data
    - Write to Transmit
    - Read to Receive

UART Flag Register

- UART0
  - UARTFR
    - Address: 0x4000C018



31		8
Reserved		

7	6	5	4	3	2	1	0
TxFE	RxFF	TxFF	RxFE	BUSY	Reserved		CTS

- Details
  - Transmit FIFO Full (TxFF)
    - 0 – Transmitter Not Full
    - 1 – Transmit Holding Register Full
  - Receive FIFO Empty (RxFE)
    - 0 – Receiver Not Empty
    - 1 – Receive Holding Register Empty

Constants

- Constants can be defined in the beginning of your assembly language program and recalled later in the program.
- Advantages
  - Ease
  - Readability
  - Modifications
- Defining Constants
  - *Constant* .equ *Value*
  - equ is a reserved word
  - Example
    - U0LSR: .equ 0x18
- Using Constants
  - Treat the constant as immediate data
  - Process to Load a 32-bit Value Into Base Register
    - MOV
      - ✓ Loads up to 16 bits into the register
      - ✓ Load lower half
    - MOVT
      - ✓ Loads 16 bits into upper half of register
      - ✓ Lower half unaffected

- Example
    - MOV   r0, =0xC000
    - MOVT r0, 0x4000
    - LDRB r1, [r0, #U0LSR]

    - 0x4000C000 is the base address
    - U0LSR is the offset

References

- Kris Schindler, *Introduction to Microprocessor Based Systems Using the ARM Microprocessor*, Second Edition, Pearson, 2013
- Muhammad Ali Mazidi, Shujen Chen, Sarmad Naimi, Sepehr Naimi, *Programming ARM Corect-M4 TM4C123G with C*, First Edition, MicroDigitalEd, 2014-2016
- Texas Instruments Incorporated, *Tiva™ TM4C123GH6PM Microcontroller Data Sheet*, June 12, 2014, Texas Instruments – Production Data, 2007-2014
- http://www.arm.com