Pomona College

Computer Science

Fall 2024: CSCI 181RT

Real-Time Systems in the Real World

Lecture 18

Thursday, October 29, 2024
Edmunds Hall 105
2:45 PM - 4:00 PM

Professor Jennifer DesCombes

Pomona College

Agenda

- Go Backs
- Discussion on Reading
- Still More Interrupts and OS Support
- Lab #9 Review
- Look Ahead
- Assignment
- Action Items

Go Backs

- General?
- Action Item Status
  - AI240910-2: Find recommended book on computer architecture.
  - AI240924-1: At what point as a development team grows does it make sense to have dedicated software and integration testers?
  - AI241024-1: Provide documentation on how to disable compiler optimization.

Pomona College

Discussion on Reading

- The Soul Of A New Machine
  - Prologue, Chapter 1 and 2: How to Make a Lot of Money, The Wars.

Interrupts and OS Support

- OS Hardware Interrupts - Time and Timers ✓
- Peripheral Hardware Interrupts
  - Input Compare (IC) ✓
  - Serial Port (UART, USART)

Interrupts and OS Support - Serial Devices

- Will be Using Single Device Architecture for Lecture/Discussion
- Simple Command and Acknowledgement Machine Control
- User Input and Output - Terminal (keyboard & monitor - CRT?)

Interrupts and OS Support - Serial Devices

- Simple Command and Acknowledgement Machine Control

```
sem_t uartTxNotBusy;
sem_t uartRxHasData;
char commandChar;
char responseChar;
char getCommand( void );
void processResponse( char );

.
.
   int myRet;
   myRet = sem_wait( &uartTxNotBusy );
   commandChar = getCommand( );
   writeUART( commandChar );
   myRet = sem_wait( &uartRxHasData );
   responseChar = readUART( );
   processResponse( responseChar );
.
.
```

Interrupt will occur when not transmitting and Tx buffer is empty

Interrupt will occur when a byte has been received.

Wait, if necessary, for the Tx portion of the USART to complete all prior transactions.

Wait, if necessary, for the Rx portion of the USART to have received a character.

NOTE: Actual system design should include some sort of timeout on the acknowledgement .

Interrupts and OS Support - Serial Devices

- Simple Command and Acknowledgement Machine Control

Tx Complete Interrupt

```
int myRet;
myRet = sem_post( &uartTxNotBusy );
goto( RESCHEDULE );
```

Interrupt will occur when not transmitting and Tx buffer is empty

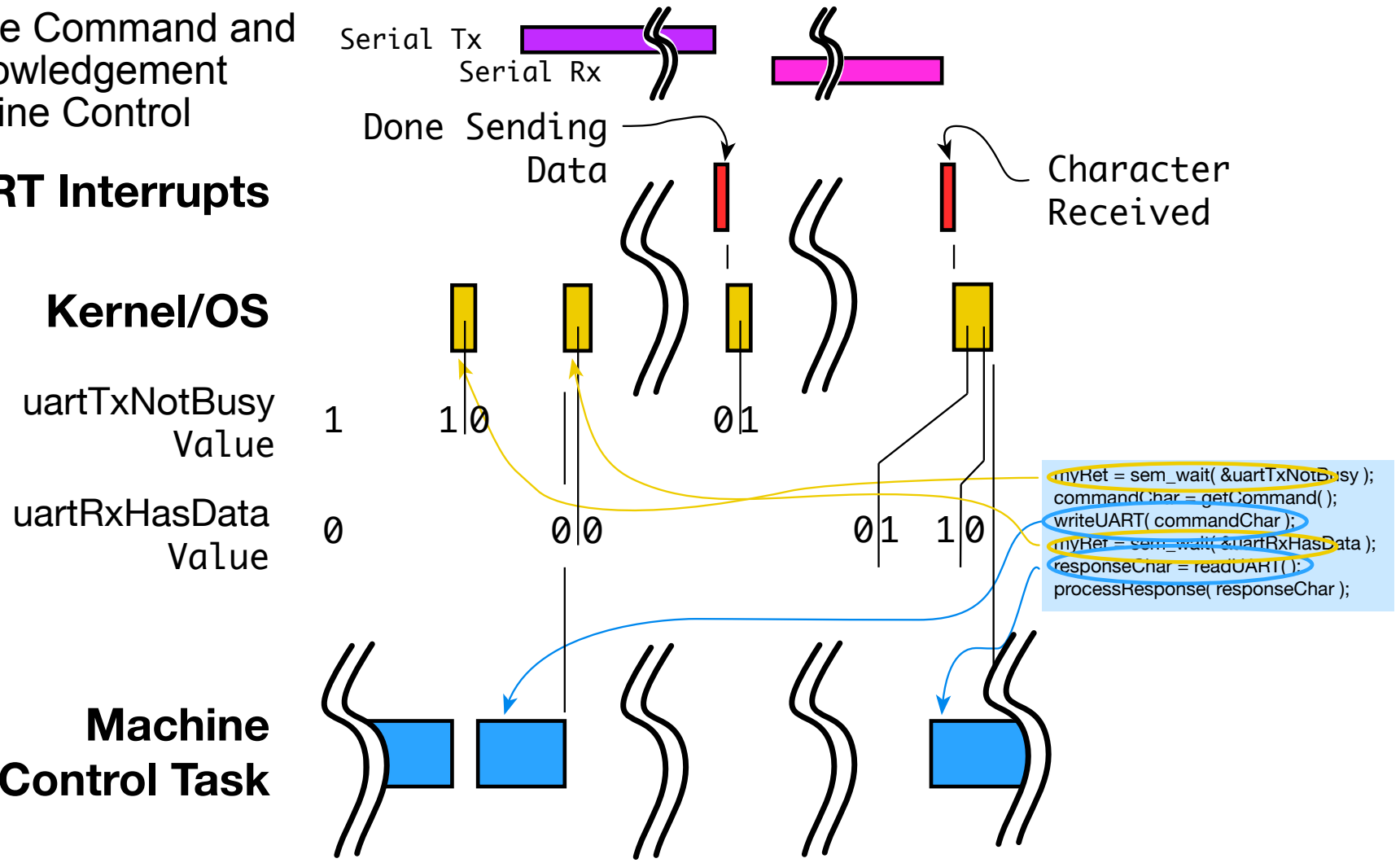
Rx Char Received Interrupt

```
int myRet;
myRet = sem_post( &uartRxHasData );
goto( RESCHEDULE );
```

Interrupt will occur when a byte has been received.

Interrupts and OS Support - Serial Devices

- Simple Command and Acknowledgement Machine Control



Interrupts and OS Support - Serial Devices

- User Input and Output (example uses character output)

```
sem_t uartTxNotBusy;
char arrayToSend[256];
int count;
.
.
   int myRet;
   for ( count = 0, count < charsToSend, count++ )
   {
      myRet = sem_wait( &uartTxNotBusy );
      writeUART( arrayToSend[ count ] );
   }
.
.
```

Interrupt will occur when not transmitting and Tx buffer is empty

Wait, if necessary, for the Tx portion of the USART to complete all prior transactions.

Interrupts and OS Support - Serial Devices

- User Input and Output (example uses character output)

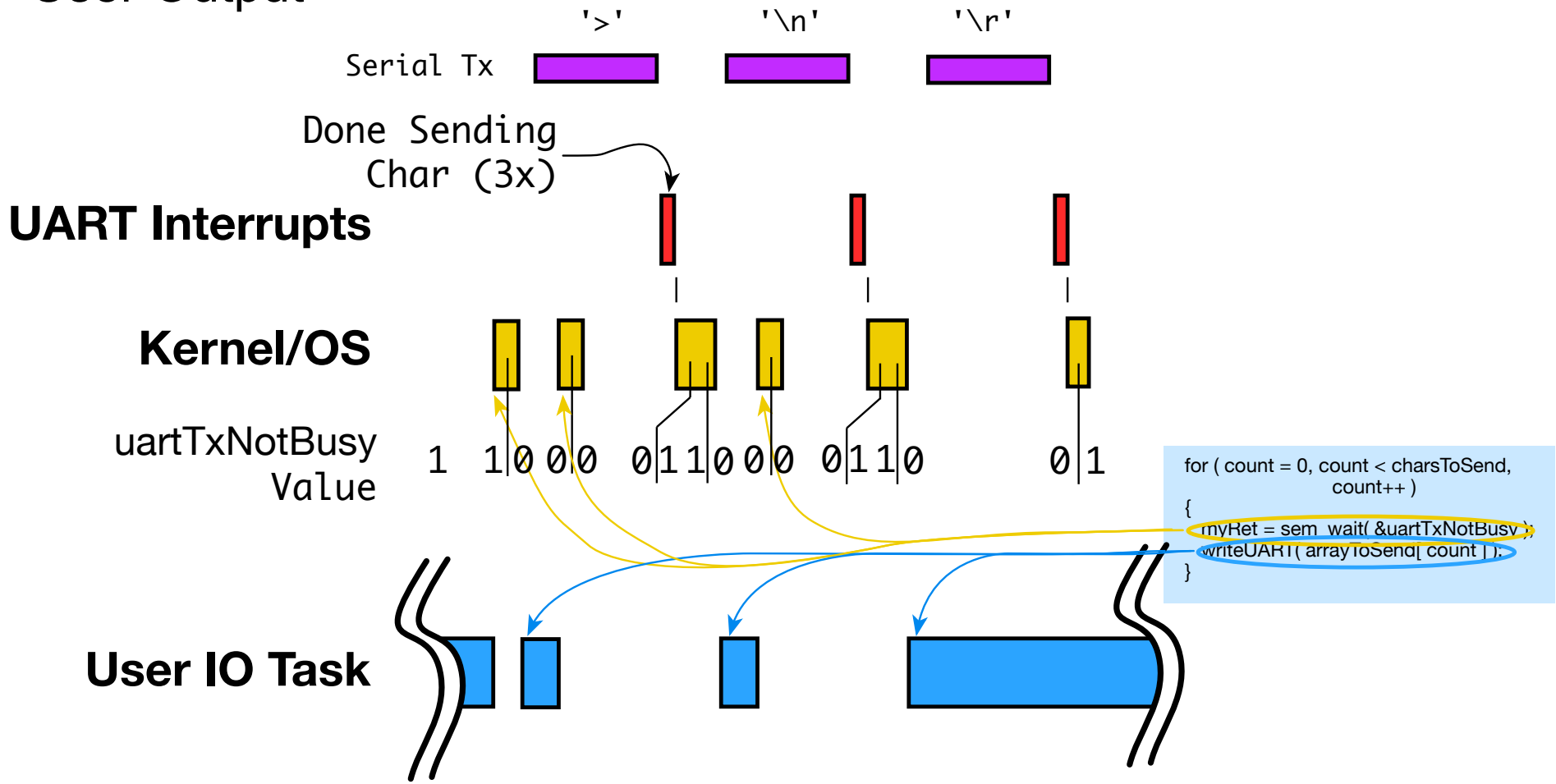
Tx Complete Interrupt

```
int myRet;
myRet = sem_post( &uartTxNotBusy );
goto( RESCHEDULE );
```

Interrupt will occur when not transmitting and Tx buffer is empty

Interrupts and OS Support - Serial Devices

- User Output



Pomona College

Interrupts and OS Support - Serial Devices

- Faster User Input and Output (example uses character output)

```
sem_t uartTxNotBusy;
char arrayToSend[256];
int count;
void trapUARTWriteBuf ( *char );
.
.
  int myRet;
  myRet = sem_wait( &uartTxNotBusy );
  trapUARTWriteBuf( &arrayToSend );
.
.
```

Interrupt will occur when not transmitting and Tx buffer is empty

Wait, if necessary, for the Tx portion of the USART to complete all prior transactions.

Perform a Software 'Trap' to Execute Code In Supervisor/ Interrupt State

Pomona College

Interrupts and OS Support - Serial Devices

- User Input and Output (example uses character output)
- Determine End of String using '\r'

Software Trap 'trapUARTWriteBuf'

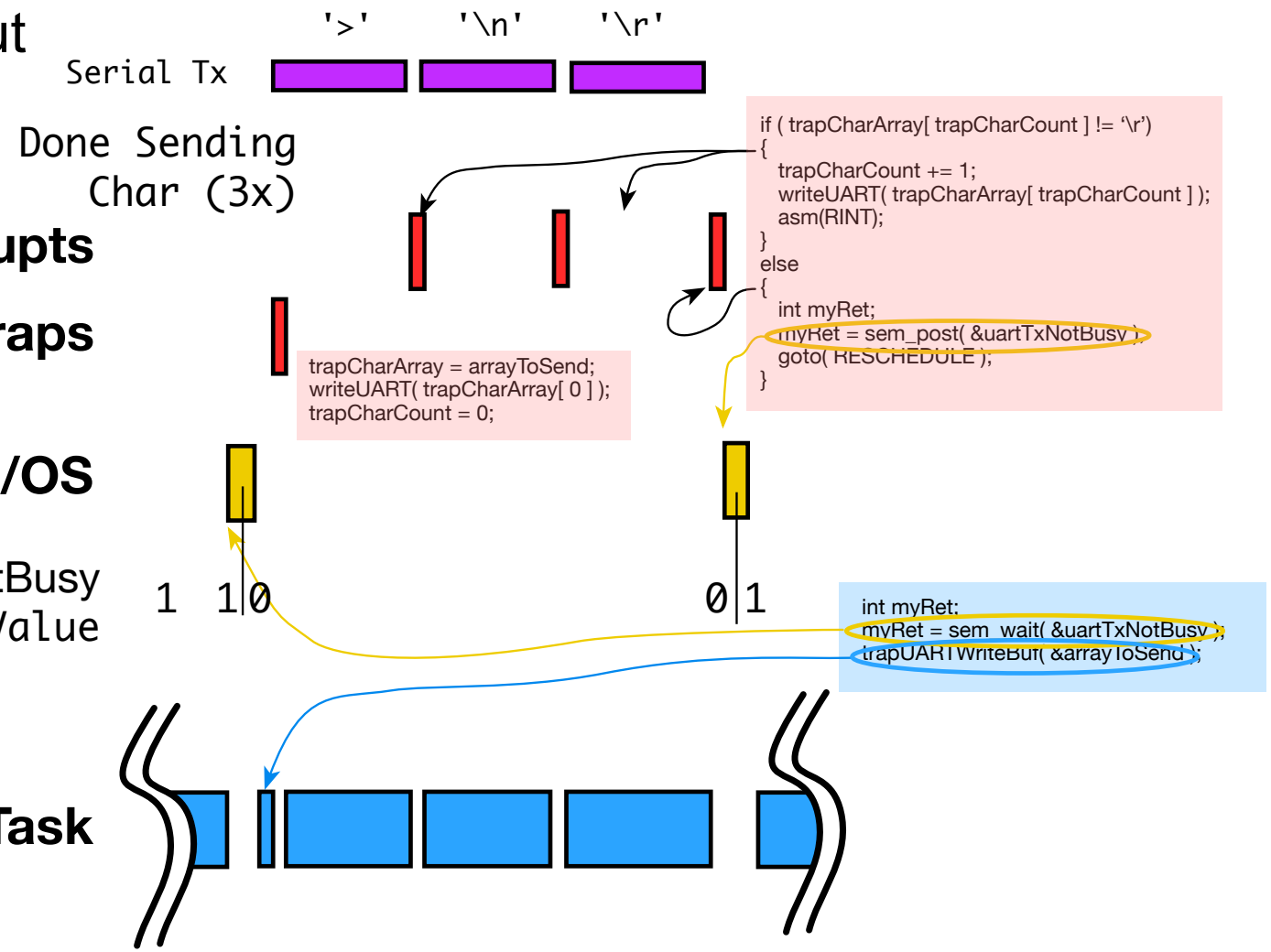
```
sem_t uartTxNotBusy;
char arrayToSend[256];
int trapCharCount;
char *trapCharArray;
.
void trapUARTWriteBuf ( *char arrayToSend )
{
   trapCharArray = arrayToSend;
   writeUART( trapCharArray[ 0 ] );
   trapCharCount = 0;
}
```

Tx Complete Interrupt

```
if ( trapCharArray[ trapCharCount ] != '\r')
{
   trapCharCount += 1;
   writeUART( trapCharArray[ trapCharCount ] );
   asm(RINT);
}
else
{
   int myRet;
   myRet = sem_post( &uartTxNotBusy );
   goto( RESCHEDULE );
}
```

Interrupts and OS Support - Serial Devices

- User Output (Faster)



Lab #9 Preview

- Incorporation of New Framework
- Optimization Must Be Removed
  - Loops Removed
  - Entire Subroutine Calls Removed
- Goals for Lab (from Lab 7 & 8)
  - Read Digital Input (GPIO1, Connector 501-Pin 5, Processor RK4)
  - Drive LED to Match Digital Input
- Sampling Rate and Data Input Rate (from Lab 7 & 8)
  - Use Function Generator to Experiment

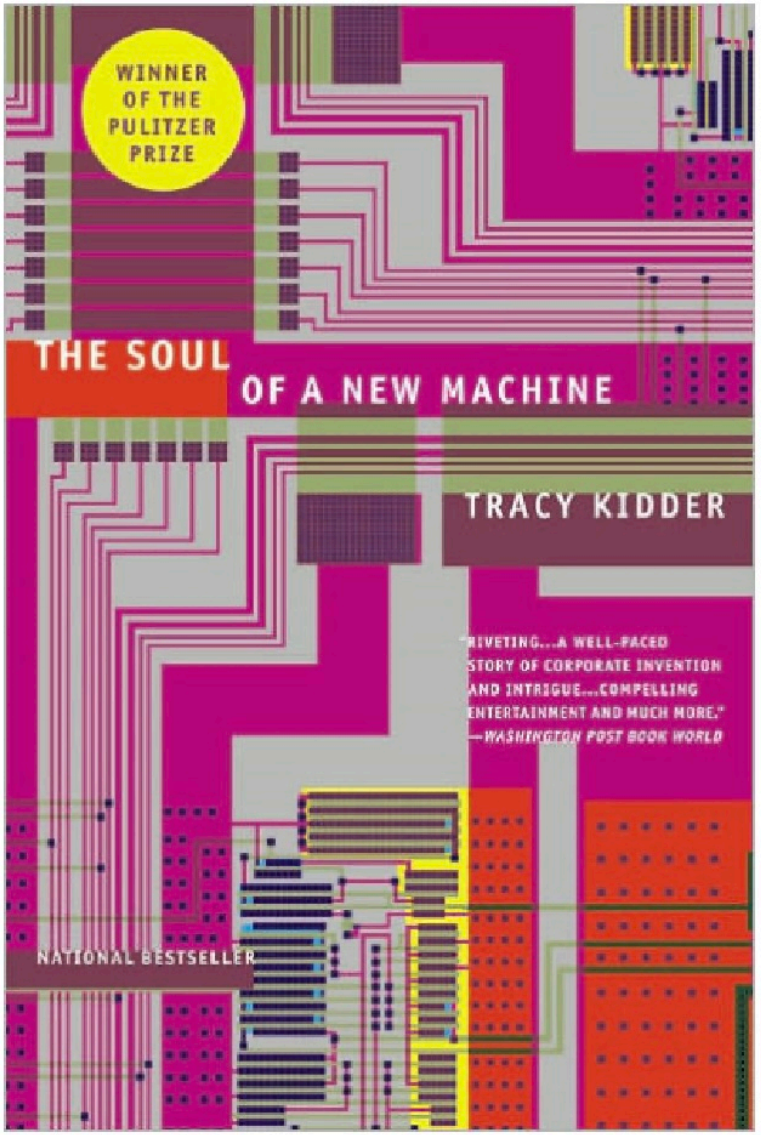
Look Ahead

- Discussion of Code Framework
- Review of Lab 9
- Discussion of Hardware Features and Capabilities - OS Related

Pomona College

Assignment - Readings

- The Soul Of A New Machine
  - None for Thursday, October 31, 2024



Assignment - Code Review

- Application Description
- Template Application
  - What Didn't You Like?
  - Why Did You Do That?
  - How Come?
- Have Comments Ready for Class on Thursday, October 31, 2024
- Feedback Will be Incorporated into Template

Assignment - OS Sequence Drawing

- Generate a OS Sequence Drawing for User Input
  - Determine End-of-string from ‘\r’
  - Can be Similar to Basic (page 10) or Faster (page 12)

Pomona College

Action Items and Discussion

AI#:	Owner	Slide #	Document	Action