Pomona College

Computer Science

Fall 2024: CSCI 181RT

Real-Time Systems in the Real World

Lecture 11

Tuesday, October 1, 2024
Edmunds Hall 105
2:45 PM - 4:00 PM

Professor Jennifer DesCombes

Agenda

- Go Backs
- Discussion on Reading
- Real Time Task Review
- Use of Semaphores (Part 1)
- Lab Overview
- Look Ahead
- Assignment
- Action Items

Pomona College

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?
  - AI240924-2: Is there a limit on the size of an Agile development effort before it becomes less efficient than other development approaches?
  - AI240924-3: Are ‘C’ type Handles (pointer to a pointer) similar in concept to Java Script Handles
  - AI240926-1: Fix the Charts - Provide Missing Information and Add Review

Pomona College

Go Backs

- Discussion on Reading - The Mythical Man Month

Real Time Task Review

- What Defines a Task
  - Endless Loop [while (true), do (forever), etc.]
  - Timing and Start / Stop Controlled by OS Calls and Events
  - Dedicated Memory Area - Stack / Heap Structure

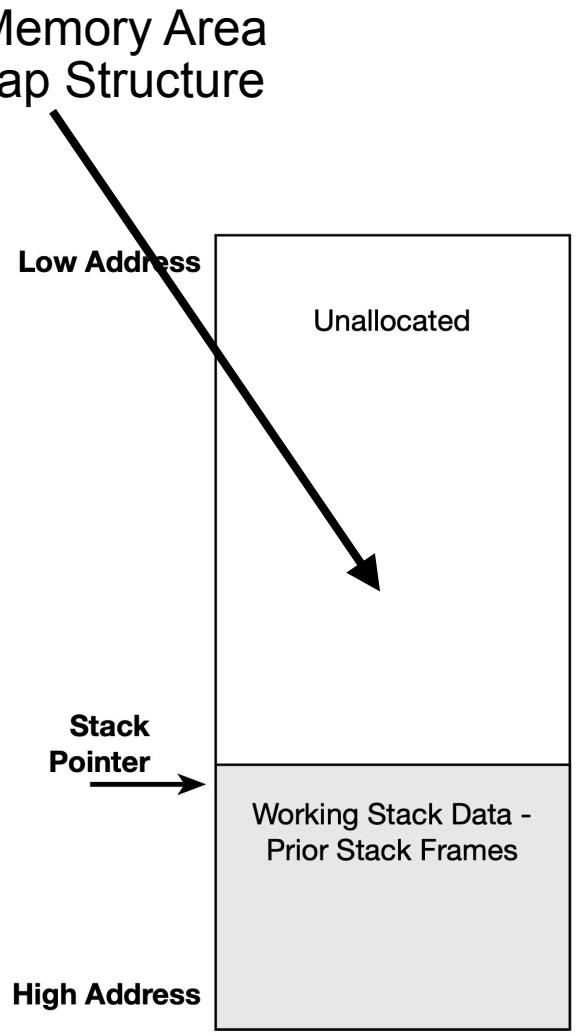
What Defines a Task

- Simple Serial Port Monitoring Task

```
// Serial Port task

#include myOSCalls.h
#include mySerialPort.h

#define true 1

while(true) {

   // Process serial port
   uartChar = readUART( );
   If (uartChar != 0)
   {
      processChar( uartChar );
   }
   myOSSleepms(5);

}
```

Endless Loop

Timing and Start / Stop Controlled by OS Calls and Events

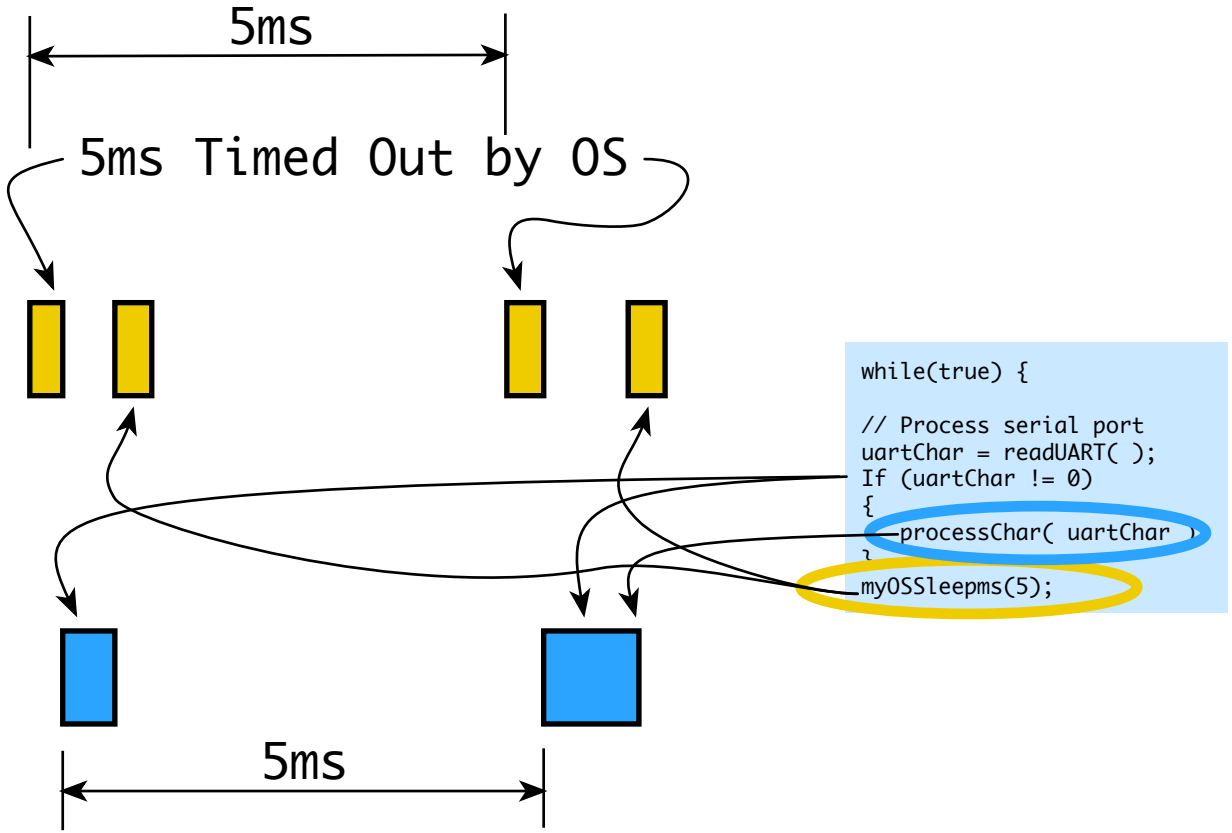


What Defines a Task

- Simple Serial Port Monitoring Task

Kernel/OS/ Interrupts

Serial Task



What Defines a Task

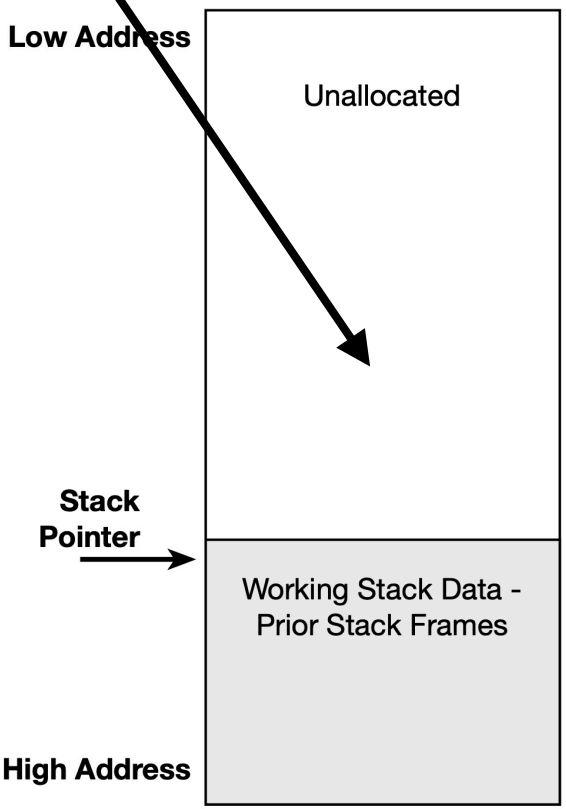
- Simple Low-priority Heartbeat Task

```
// Heartbeat task

#include myOSCalls.h
#include myIOAbstraction.h

#define true 1

while(true) {

    // turn on and off heartbeat
    // LED at a half-second rate
    myIOHeartbeatLEDOn();
    myOSSleepms(1000);
    myIOHeartbeatLEDOff();
    myOSSleepms(1000);

}
```

Endless Loop

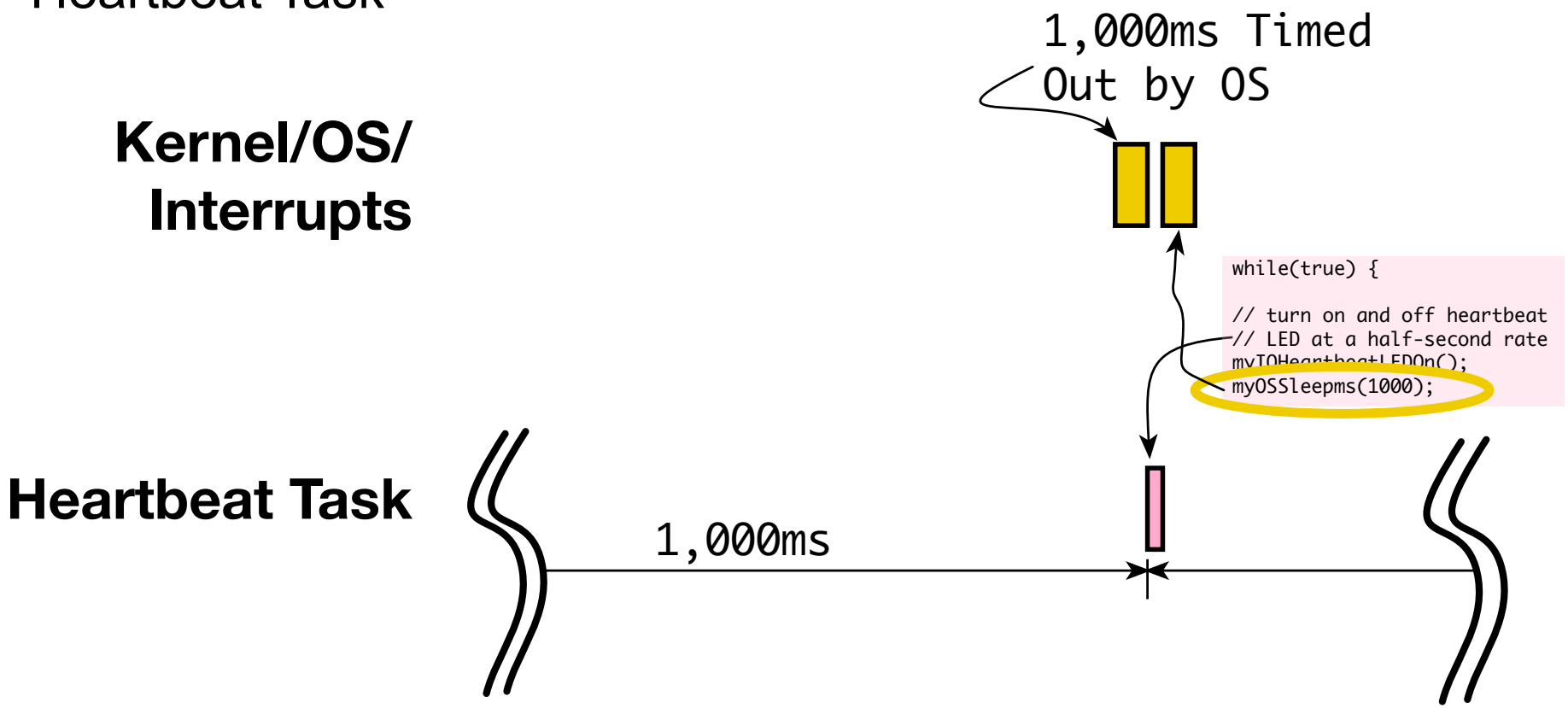
Timing and Start / Stop Controlled by OS Calls and Events

Dedicated Memory Area
- Stack / Heap Structure



What Defines a Task

- Simple Low-priority Heartbeat Task



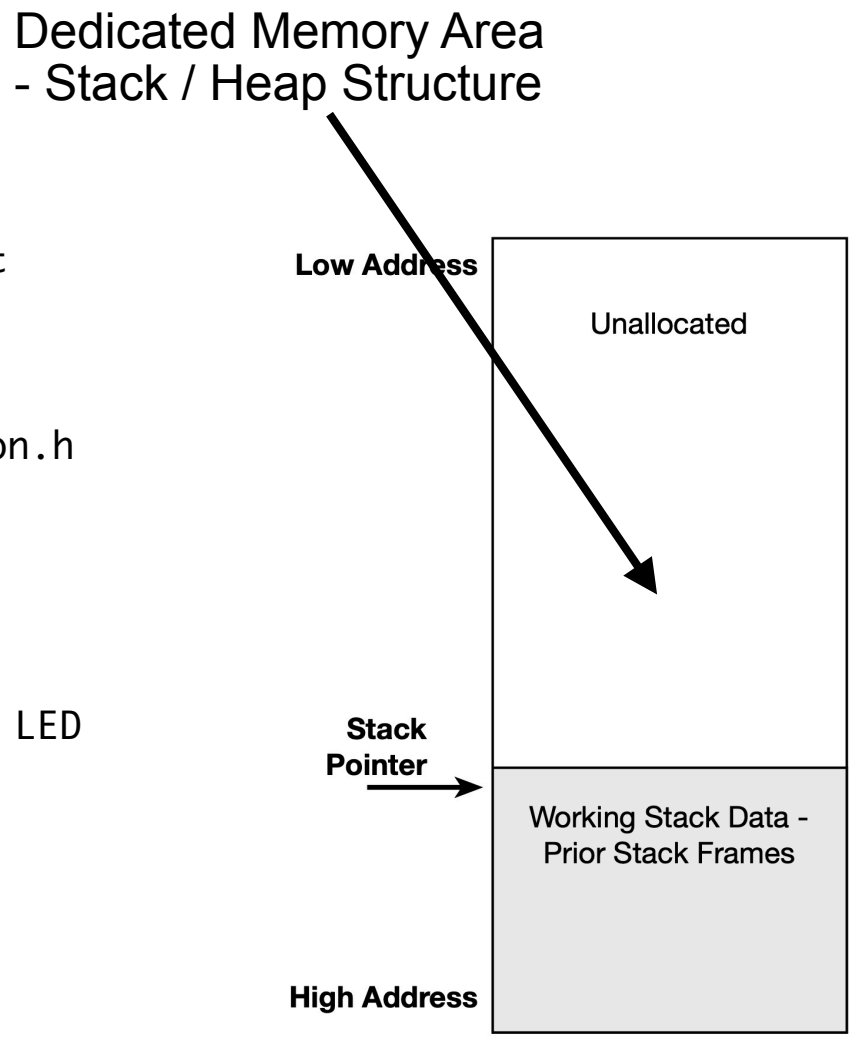
What Defines a Task

- Dummy / Always Runnable Task

```
// Dummy Task
// NOTE: Must be lowest
//   priority task

#include myOSCalls.h
#include myIOAbstraction.h

#define true 1

while(true) {

    // Constantly flash LED
    myIODummyLEDOn( );
    myIODummyLEDOff( );

}
```

Endless Loop

There are no Timing and Start / Stop OS Calls or Events



What Defines a Task

- Dummy / Always Runnable Task

Kernel/OS/ Interrupts

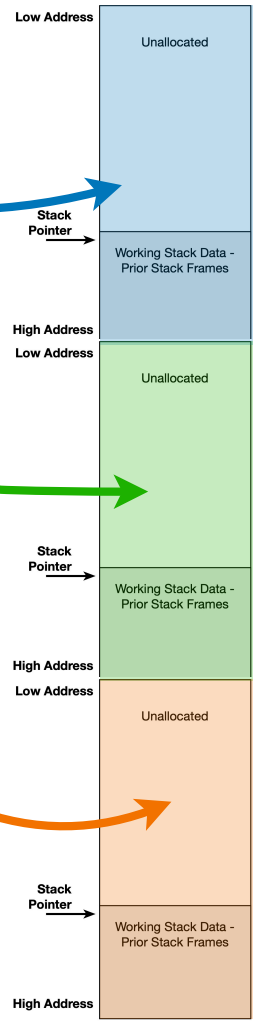
Dummy Task

System With Three Tasks

```
// Heartbeat task

#include myOSCalls.h
#include myIOAbstraction.h

#define true 1

while(true) {

    // turn on and off heartbeat
    // LED at a half-second rate
    myIOHeartbeatLEDOn();
    myOSSleepms(1000);
    myIOHeartbeatLEDOff();
    myOSSleepms(1000);

}
```

```
// Dummy Task
// NOTE: Must be lowest
//   priority task

#include myOSCalls.h
#include myIOAbstraction.h

#define true 1

while(true) {

    // Constantly flash LED
    myIODummyLEDOn( );
    myIODummyLEDOff( );

}
```

```
// Serial Port task

#include myOSCalls.h
#include mySerialPort.h

#define true 1

while(true) {

    // Process serial port
    uartChar = readUART( );
    If (uartChar != 0)
    {
       processChar( uartChar );
    }
    myOSSleepms(5);

}
```



What's a *main* To Do?

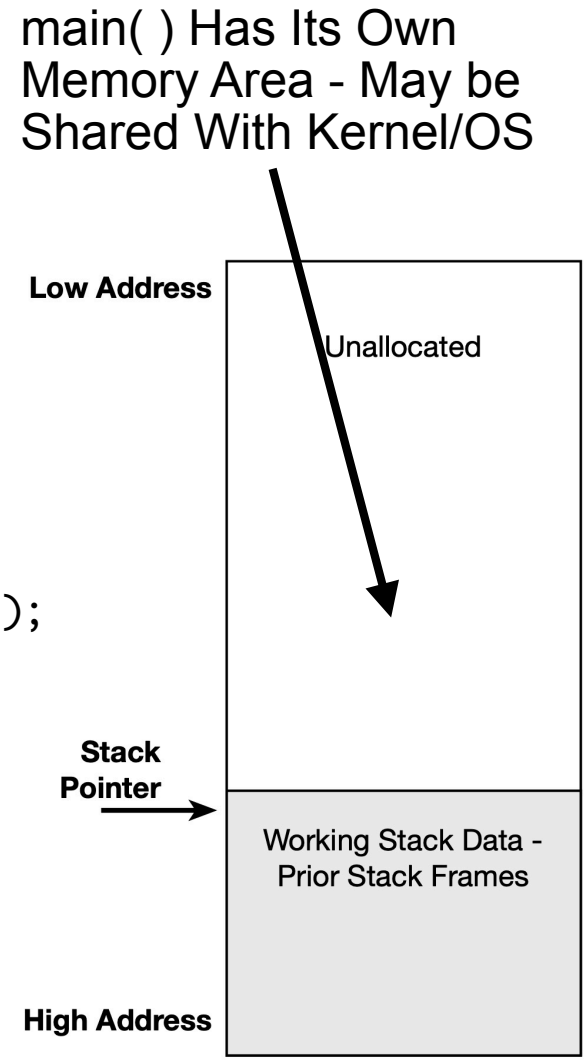
- main( ) Creates, Initializes, Starts, and Thats All!

```
// Main Program

#include myOSCalls.h
#include heartbeatTask.h
#include serialTask.h
#include dummyTask.h

int main( )
{

   osCreate( &serialTask, 1 );
   osCreate( &heartbeatTask, 2 );
   osCreate( &dummyTask, 3 );

   initSerialTask( );
   initHeartbeatTask( );
   initDummyTask( );

   osStartTasks( );
   // Never returns

}
```

Creates Tasks Within RTOS and Assign Priority

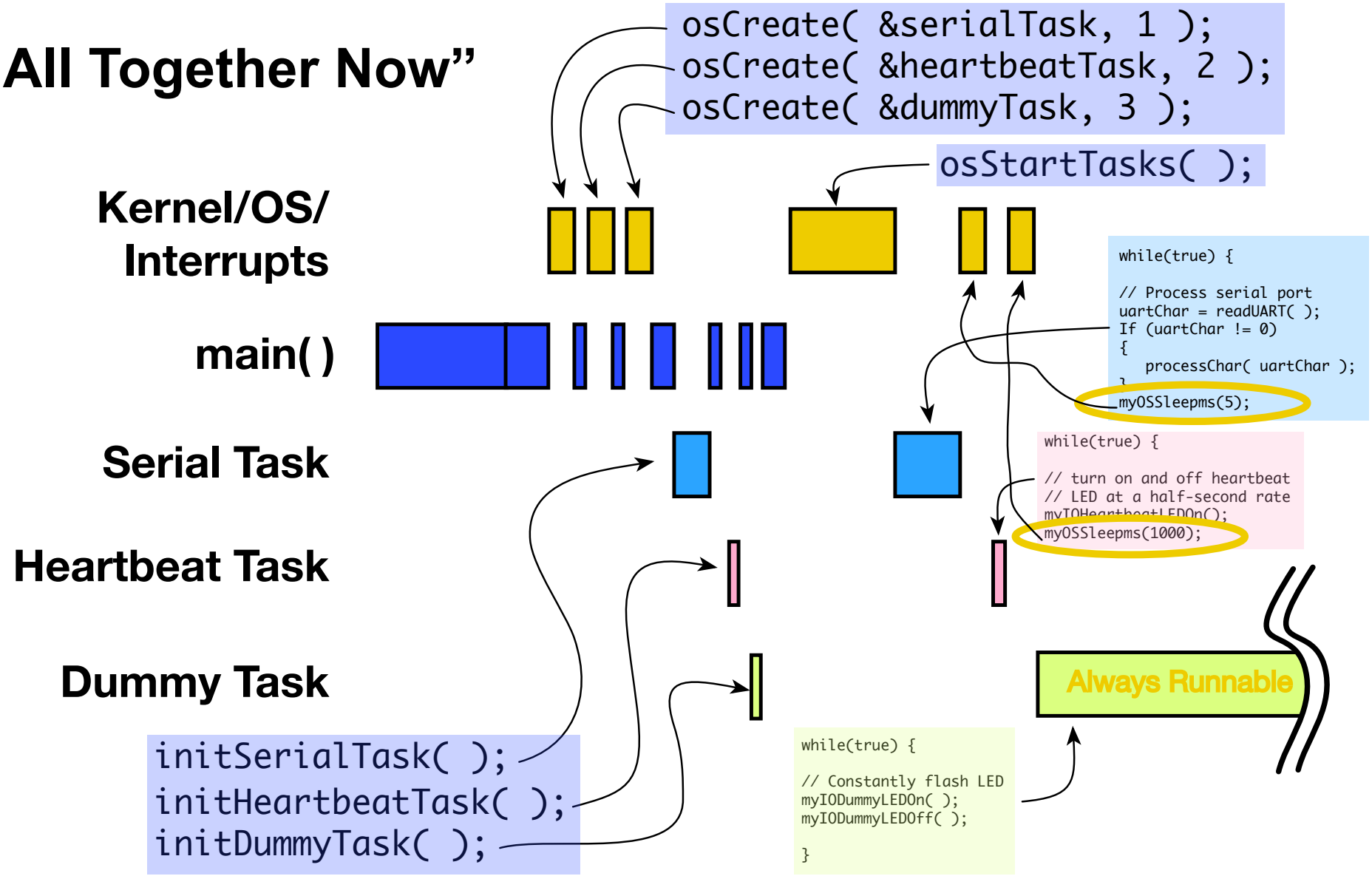
Call Task Initialization Prior to Starting Task

Start Tasks Running

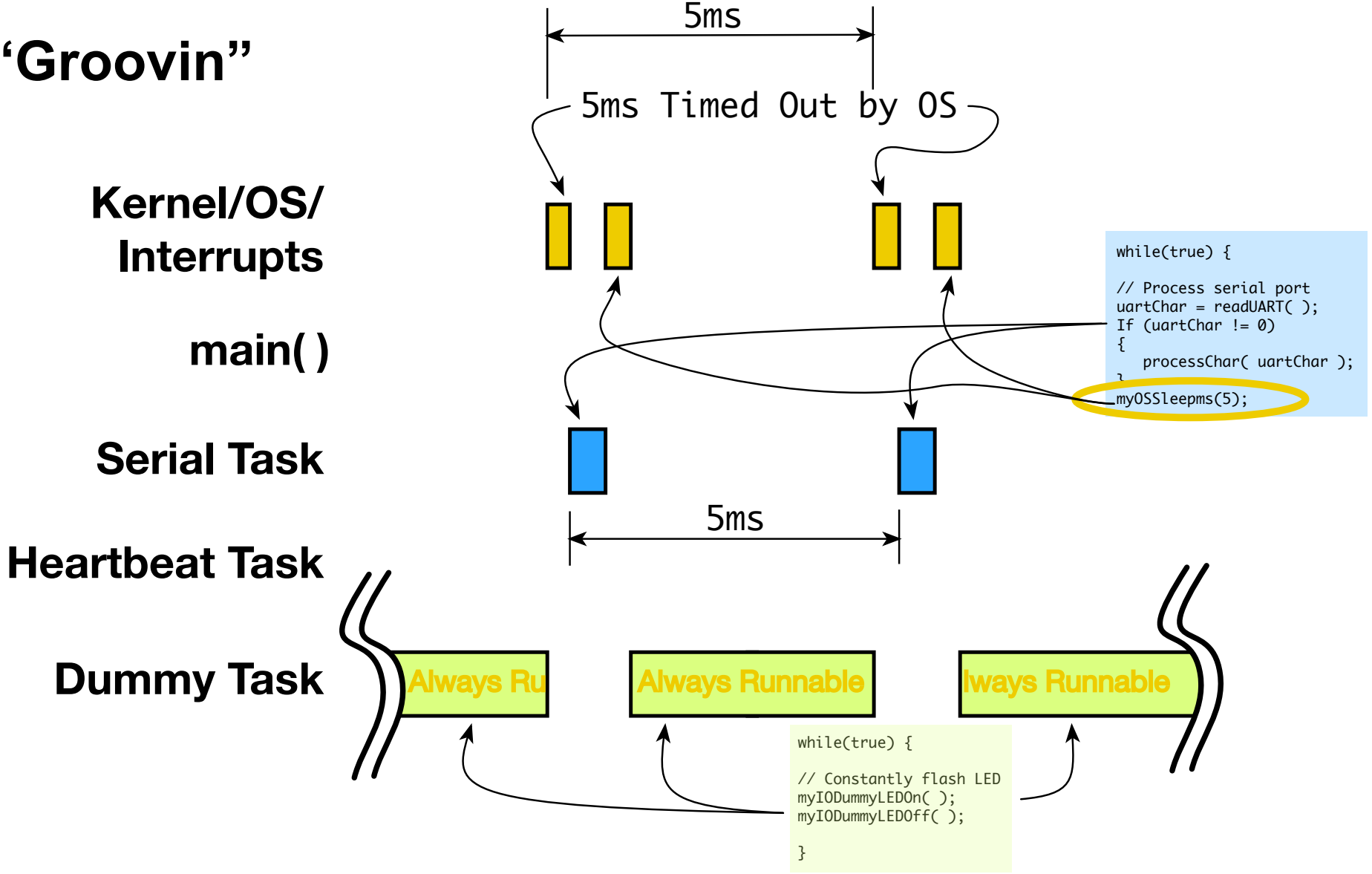


Pomona College

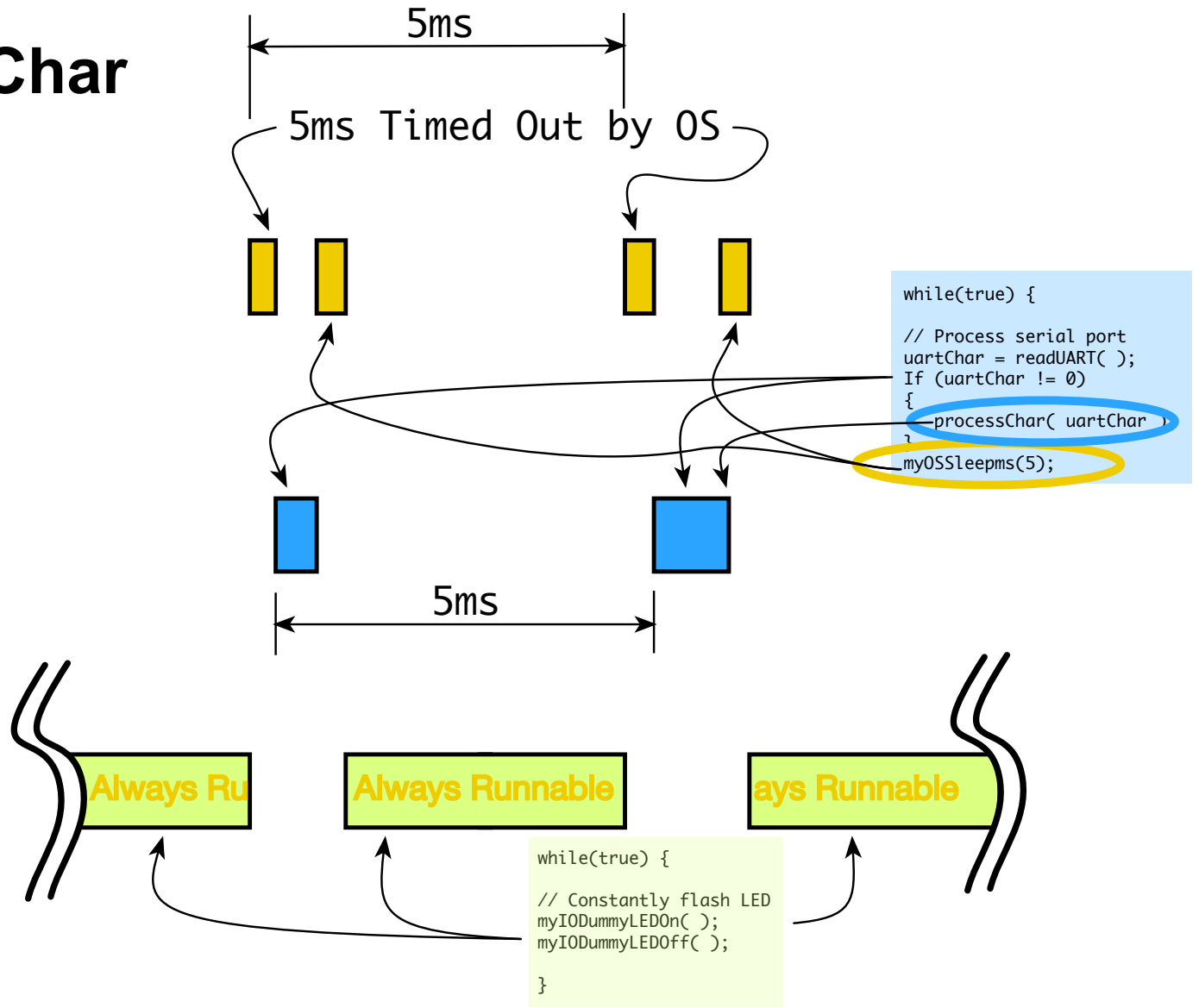
"All Together Now"



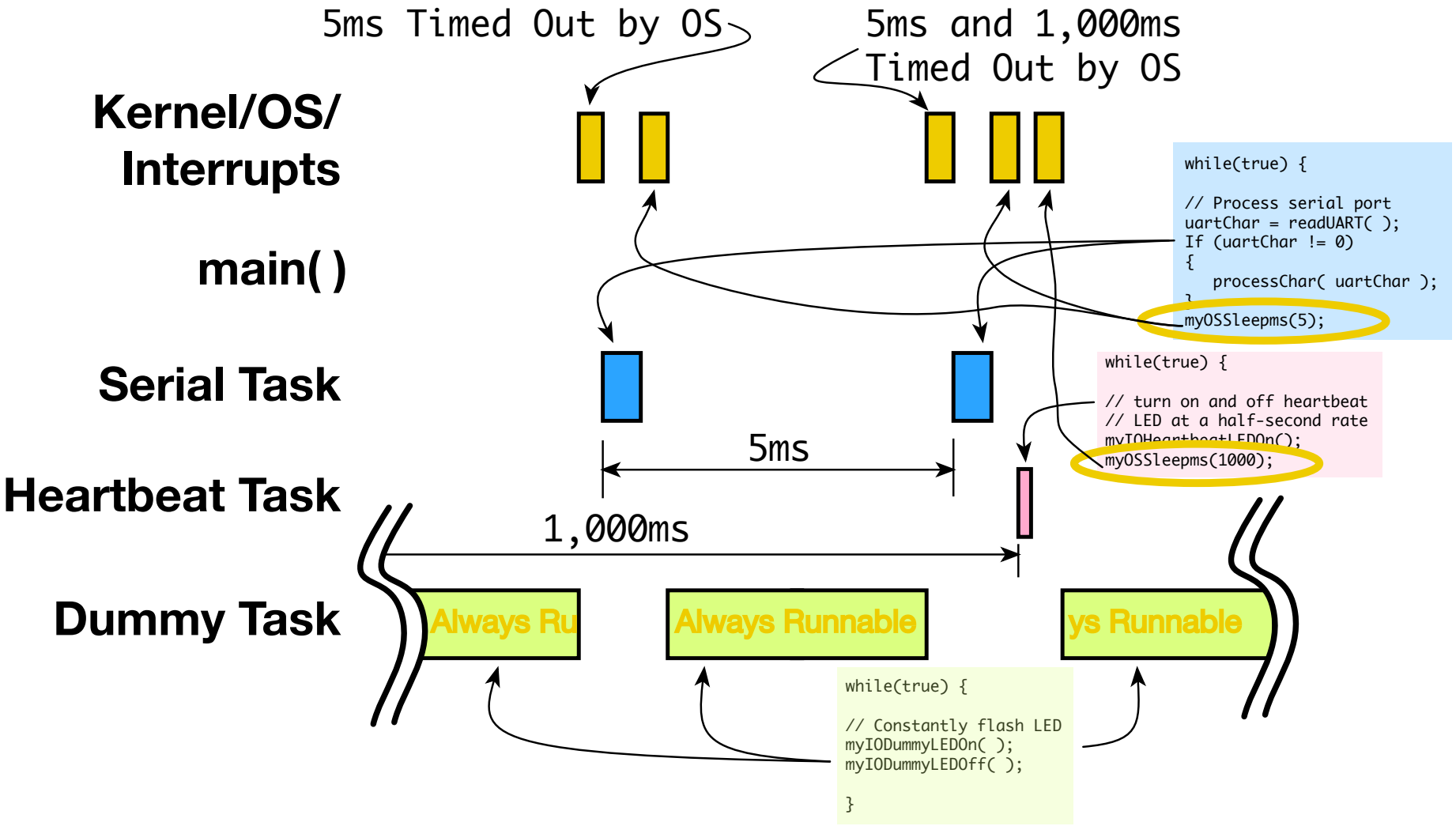
"Groovin"


5ms
5ms Timed Out by OS
Kernel/OS/
Interrupts
main( )
Serial Task
Heartbeat Task
Dummy Task
5ms
Always Ru
Always Runnable
lways Runnable
while(true) {
// Process serial port
uartChar = readUART( );
If (uartChar != 0)
{
processChar(uartChar);
}
myOSSleepms(5);
while(true) {
// Constantly flash LED
myIODummyLEDOn( );
myIODummyLEDOff( );
}

"Groovin" + Char

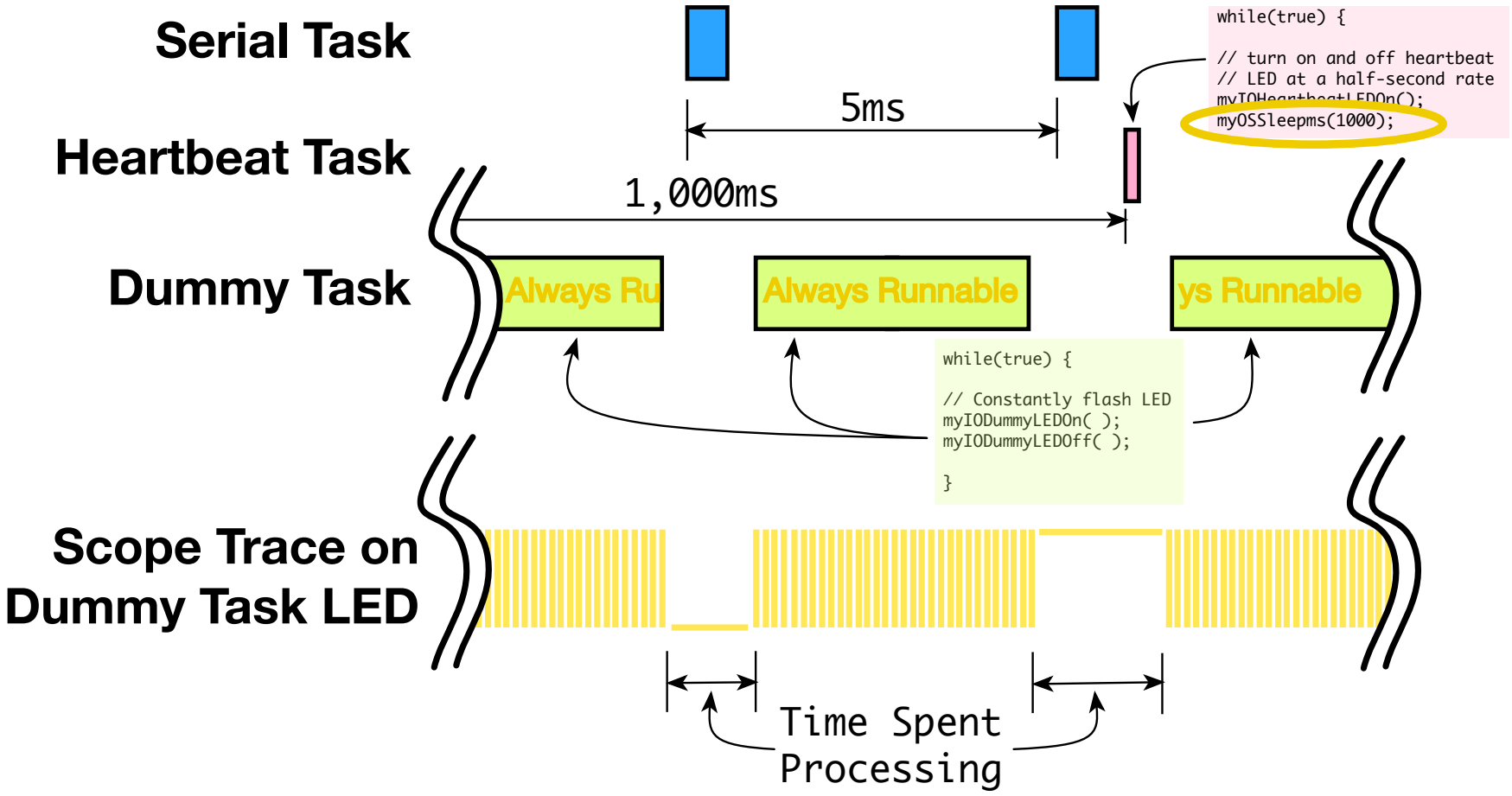


"Just the Two of Us" - Heartbeat Task Executes



"Just the Two of Us" - Heartbeat Task Executes

- Use Oscilloscope to Monitor Spare Processing Time



Task Structure Review

- Synchronization of Tasks
- Protection of Shared Devices/Services
- Controlled Processing of Input Data
- Protection of Non-reentrant Code
- Guarantee Completion of Specific Operations

Use of Semaphores

- Semaphore Functions in Different RTOS

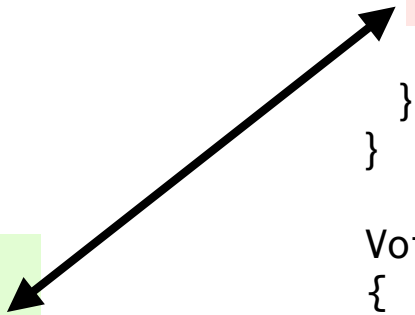
RTOS	P-Action	V-Action
POSIX	int **sem_wait**(sem_t *sem);	int **sem_post**(sem_t *sem);
FreeRTOS	pdStatus **xSemaphoreTake**(SemaphoreHandle_t xSemaphore, TickType_t xTicksToWait);	pdStatus **xSemaphoreGive**(SemaphoreHandle_t xSemaphore);
VxWorks	STATUS **semTake**(SEM_ID semId, int timeout);	STATUS **semGive**(SEM_ID semId);

- Code Snippets Not Complete
- Color Highlights Show Semaphore Control

Pomona College

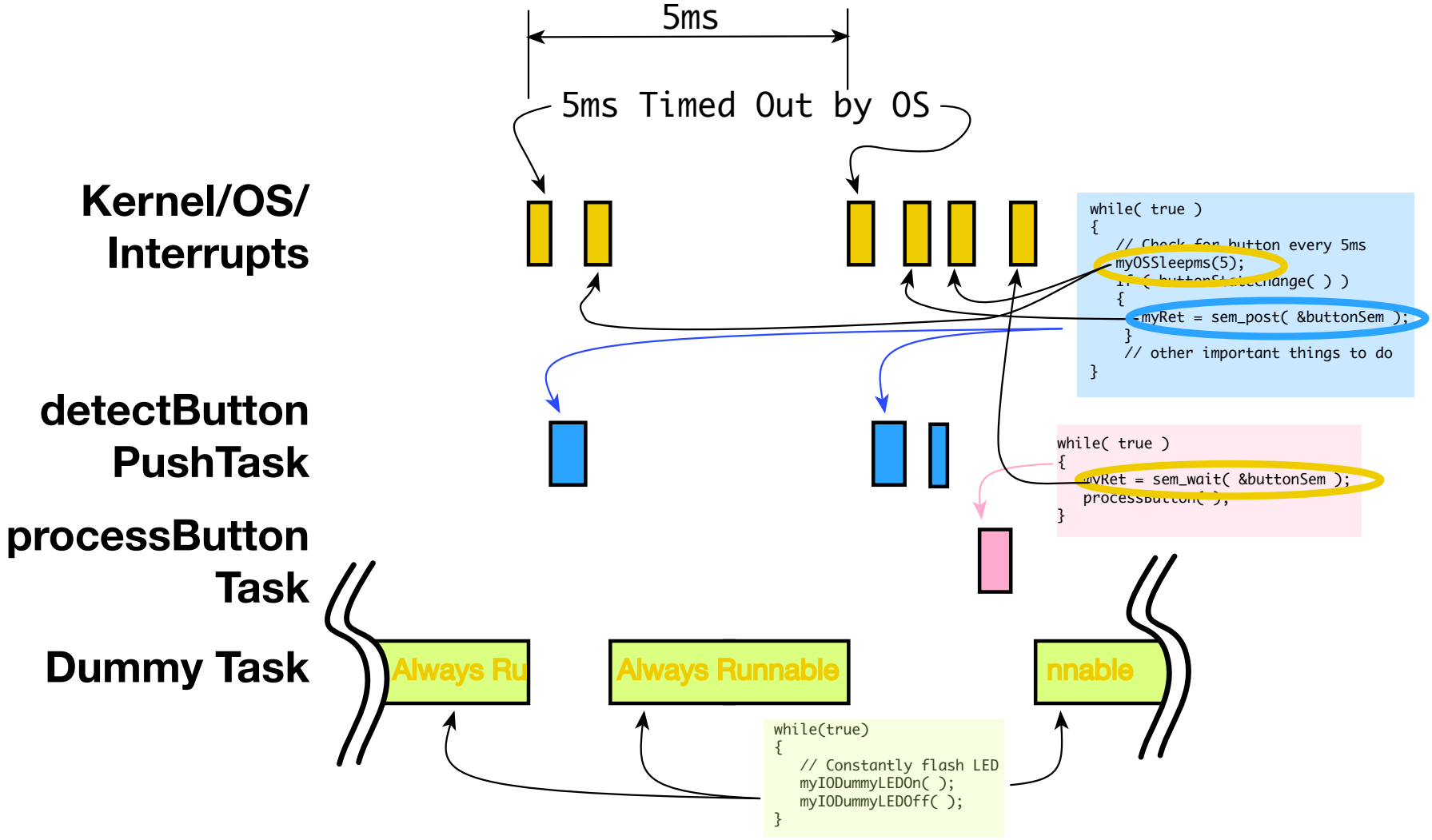
Use of Semaphores - Task Synchronization

```
sem_t buttonSem;
bool buttonStateChange( void );
void processButton( void );

// High Priority Task
void detectButtonPushTask (void)
{
   int myRet;
   while( true )
   {
      // Check for button every 5ms
       myOSSleepms(5);
      If ( buttonStateChange( ) )
      {
        myRet = sem_post( &buttonSem );
      }
      // other important things to do
   }
}
```



```
// Lower Priority Task
void processButtonTask (void)
{
   int myRet;
   while( true )
   {
      myRet = sem_wait( &buttonSem );
      processButton( );
   }
}

Void dummyTask( void )
{
    while(true)
    {
        // Constantly flash LED
        myIODummyLEDOn( );
        myIODummyLEDOff( );
    }
}
```

Use of Semaphores - Task Synchronization



Lab 5 Goals

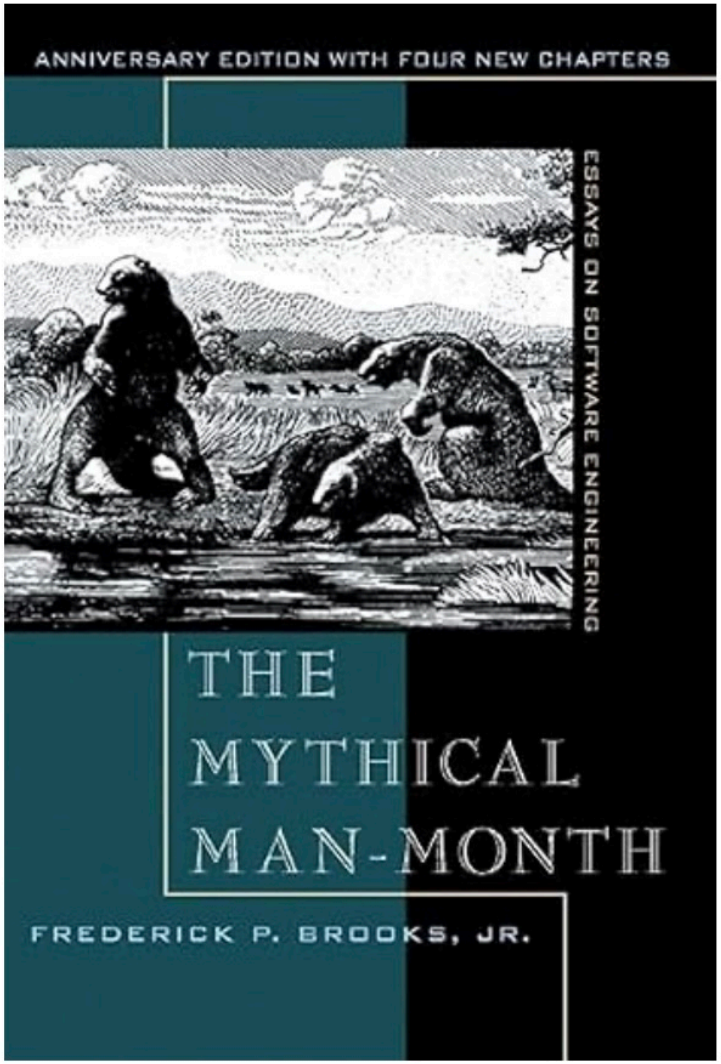
- Complete Any Remaining PWM Software
- Add “DummyTask” Like LED Strobe to 5ms +/- 2ms Loop
- Measure Processing Time, Calculate Utilization

Look Ahead

- Review of Reading
- More Semaphore and Task Control (Part 2)
- Discussion of Lab 5

Assignment - Readings

- The Mythical Man Month
  - Chapter 7, 8 & 9: Why Did the Tower of Babel Fail? - Calling the Shot - Ten Pounds in a Five-Pound Sack
  - Send Me Discussion Topics by 10:00 AM on Thursday, Oct. 3, 2024.



Pomona College

Action Items and Discussion

AI#:	Owner	Slide #	Document	Action