

Startup\_rvmk.s file

EXTERN UART1IntHandler

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;

; The vector table.

;

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EXPORT \_\_Vectors

\_\_Vectors

DCD StackMem + Stack ; Top of Stack

DCD Reset\_Handler ; Reset Handler

DCD NmiSR ; NMI Handler

DCD FaultISR ; Hard Fault Handler

DCD IntDefaultHandler ; The MPU fault handler

DCD IntDefaultHandler ; The bus fault handler

DCD IntDefaultHandler ; The usage fault handler

DCD 0 ; Reserved

DCD 0 ; Reserved

DCD 0 ; Reserved

DCD 0 ; Reserved

DCD vPortSVCHandler ; SVCall handler

DCD IntDefaultHandler ; Debug monitor handler

DCD 0 ; Reserved

DCD xPortPendSVHandler ; The PendSV handler

DCD xPortSysTickHandler ; The SysTick handler

DCD IntDefaultHandler ; GPIO Port A

DCD IntDefaultHandler ; GPIO Port B

DCD IntDefaultHandler ; GPIO Port C

DCD IntDefaultHandler ; GPIO Port D

DCD IntDefaultHandler ; GPIO Port E

DCD IntDefaultHandler ; UART0 Rx and Tx

DCD UART1IntHandler ; UART1 Rx and Tx

Void Uart\_init(){

// Enable UART1

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_UART1);

// Enable PortC

ROM\_SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOC);

// Configure the pin multiplexing

ROM\_GPIOPinConfigure(GPIO\_PC4\_U1RX);

ROM\_GPIOPinConfigure(GPIO\_PC5\_U1TX);

// Configure the type of the pins for UART Tx/Rx

ROM\_GPIOPinTypeUART(GPIO\_PORTC\_BASE, GPIO\_PIN\_4|GPIO\_PIN\_5);

ROM\_UARTConfigSetExpClk(UART1\_BASE, ROM\_SysCtlClockGet(), 9600,

(UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE |UART\_CONFIG\_PAR\_NONE));

UARTFIFODisable(UART1\_BASE);

// Enable the UART1 interrupt handler

ROM\_IntEnable(INT\_UART1);

ROM\_UARTIntEnable(UART1\_BASE, UART\_INT\_RX|UART\_INT\_RT );

}

void UART1IntHandler(void)

{

//Get the interrupt status and clear the asserted interrupts

ROM\_UARTIntClear(UART1\_BASE, ROM\_UARTIntStatus(UART1\_BASE, true));

while(ROM\_UARTCharsAvail(UART1\_BASE))

{

NewChar = ROM\_UARTCharGet(UART1\_BASE)&0x00FF; // Mask off MSB

}

}