```c
/********************************************************************************************************
Project : SRAM Example Test Code
Version : 2.0
Date    : 11/12/04 (9:34PM - Yes, I'm writing software on a friday night - I have no life!)
Author  : Chris Troutner
Company : MyRobot
Comments: This version is my first attempt at putting together an example program to interface with the
            32KB External SRAM chip on the MyRobot Mini-PC. This program simply uses malloc() to
            copy a string to external memory, and then display the value of string1, a pointer to that
            string, on the LCD.

Robot Type: Mini-Computer

Notes   : See a really good website on the explanation of malloc, sizeof, and free:
            http://www.eecs.harvard.edu/~ellard/Q-97/HTML/mem/node3.html

          Make sure the makefile contains the following:
            # 32 KB of external RAM, starting after internal RAM (ATmega128),
            # only used for heap (malloc()). (.data & .bss kept in SRAM)
            LDFLAGS += -Wl,--defsym=__heap_start=0x801100,--defsym=__heap_end=0x8090FF

********************************************************************************************************/
//Compiler Derectives

    //Includes
    #include <modelt.h>
    #include <stdlib.h>

    //Defines

//Global Variables


//Sub-Function Prototypes

//Main
    int main(void)
    {
    //Local Variables
        char *string1;      //Pointer to the string in external memory
        char t[16];     //Initial string holder
        char u[16];     //Final string holder

    //Initialization
        reset();            //Reset and all ports and peripherials
        lcd_init();         //Initialize the LCD

        //!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
        //*********************************
        // Initialize the External SRAM
        //*********************************
        MCUCR = (1<<SRE)|(0<<SRW10);
        //  SRE = 1 - Enable External Memory
        XMCRA = (1<<SRL2)|(0<<SRL1)|(1<<SRL0)|(1<<SRW01)|(0<<SRW00)|(0<<SRW11);
```

```
        //  SRL2,SRL1,SRL0 = 101    - Lower Sector of memory = 0x1100 to 0x9FFF (32K of XRAM) (pg. 30)
        //                            Upper Sector of Memory = 0xA000 to 0xFFFF (doesn't exist)
        //  SRW01,SRW00 = 10        - Wait two cycles durring read/write strobe (pg. 30)
        //  SRW11,SRW10 = 00        - No wait-states for accessing this memory (memory doesn't exist)
        XMCRB = (1<<XMBK)|(0<<XMM2)|(0<<XMM1)|(1<<XMM0);
        //  XMBK = 1                - External Memory Bus-Keeper Enabled (pg. 31)
        //  XMM2,XMM1,XMM0 = 001    - Release PC7 on PORTC (pg. 31)

        DDRC |= 0x80;               //Set PC7 as an output
        PORTC |= 0x80;              //Drive a high on PC7 to enable the memory

//!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

    //Main Exectution Code
        string1=(char *)malloc(sizeof(16));     //Allocate 16 characters in external memory

        // Test for NULL returned by the malloc() function
        if(string1 == NULL)
        {
            sprintf(t, "Error: NULL");              //Report an error and loop forever if
            line2(t);                               //There is no room in external SRAM
            while (1) {};
        }

        strcpy(string1,"Test String");              //Copy the string to External SRAM
//      strcpy(string1,t);

        strcpy(u,string1);                          //Copy the string into internal SRAM
        line1(u);                                   //Display the string on line1

        line2(string1);                             //Alternativly, we can call the string
                                                    //directly from external SRAM.

        free(string1);                              //When done, don't forget to free up
                                                    //space with the free() function

        while (1)                                   //Loop forever.
        {

        }

    }

//Sub-Functions
```