TI-RSLK MAX

Texas Instruments Robotics System Learning Kit

Texas Instruments

Module 4

Activity: Software Design using MSP432

Activity: Software Design using MSP432

Question 1

Write a C function that returns true if an ASCII character is a letter, and false otherwise. The letters exist from 0x41 to 0x5A and from 0x61 to 0x7A inclusive. The prototype for this function is

```
int bLetter(char data);
```

Question 2

Write a C function to calculate the average of three numbers. Assume the three numbers are passed by value into your function. The prototype for this function is

```
int32_t Average(int32_t n1, int32_t n2, int32_t n3);
```

Question 3

Write a C function to find the maximum of three numbers. Assume the three numbers are passed by value into your function. The prototype for this function is

```
int32_t Max(int32_t n1, int32_t n2, int32_t n3);
```

Question 4

Write a C function to calculate the quadratic equation

$$y = 2x^2 - 3x + 1$$

assuming x and y are 32-bit numbers. Some values of x will cause the calculation of y to extend beyond the values allowed by 32-bit signed numbers. Determine the largest possible value for x, such that $y < 2^{31}$. Use this threshold to return y = 0x7FFFFFF (2^{31}-1) if the input value would create overflow. Determine the smallest possible value for x, such that $y > -2^{31}$. Use this threshold to return y = 0x80000000 (-2^{31}) if the input value would create underflow. The prototype for this function is

```
int32_t Quadratic(int32_t x);
```

Question 5

Write a C function that calculates the square distance between two points (x1, y1) and (x2, y2)

$$d = (x1-x2)^2 + (y1-y2)^2$$

assuming x1 x2, y1, and y2 are signed 32-bit numbers. You may assume the numbers are small enough that overflow does not occur. The prototype for this function is

```
int32_t SquareDistance(int32_t x1, int32_t y1,
        int32_t x2, int32_t y2);
```

Question 6

Write a C function that returns true if $10 \le x < 99$, and false otherwise. The prototype for this function is

```
int bTwoDigit(uint32_t x);
```

Question 7

Unsigned 32-bit numbers range from 0 to 2^{32}-1 (4294967295). Write a C function that takes an unsigned 32-bit number and returns a result from 0 to 10 defining the number of decimal digits required to represent the number. For example, the input of 0 returns 0, the input of 1 – 9 returns 1, the input of 10 – 99 returns 2, etc. The prototype for this function is

```
uint32_t NumDigits(uint32_t x);
```

Question 8

Write a C function that multiplies two unsigned 32-bit numbers. Implement overflow detection such that if the product were to exceed 2^{32}-1, the function returns 0xFFFFFFFF (2^{32}-1). The prototype for this function is

```
uint32_t Product(uint32_t n1, uint32_t n2);
```

ti.com/rslk

Module 4

Quiz: Software Design using MSP432

Quiz: Software Design using MSP432

Q1 Conditional

Write a C function to find the minimum of three numbers. Assume the three numbers are passed by value into your function. The prototype is
`int16_t Min(int16_t n1, int16_t n2, int16_t n3);`

Q2 Conditional

Write a C function to returns a true if an ASCII character is a hex digit. Hex digits are 0x30 to 0x39 and 0x41 to 0x46 inclusive. The prototype is
`int isHex(char data);`

Q3 Conditional

Write a C function to returns the absolute value of a number. The input is signed, but the output will be unsigned. The prototype is
`uint32_t Abs(int32_t data);`

Q4 Calculations

Write a C function to calculate the equation

y = 1000/x – (3*x+1)/4

assuming x and y are 32-bit numbers. Return y = 0x7FFFFFFF (2^{31}-1) if the input value is zero, otherwise you can ignore overflow. The prototype for this function is
int32_t Calculate(int32_t x);

Q5 Calculations

Assume x1, x2, x3, x4 are four measurements collected at 1ms time intervals. Calculate the discrete derivative using this equation

d = x1+3*x2-3*x3-x4

If the units of x1 is mV, then the units of d will be mV/ms (or V/s). Assume the inputs are 16-bit signed numbers ranging from 0 to 3300. Solve overflow by limiting the output to -1000 to +1000 V/s. Hint, calculate the intermediate result in 32-bit math, check for overflow, and then return a 16-bit result. The prototype for this function is
`int16_t Derivative(int16_t x1, int16_t x2, int16_t x3, int16_t x4);`

IMPORTANT NOTICE AND DISCLAIMER