Application Examples

MARS & BlueShift Development Kits

Introduction

This document contains additional information to support the application examples included in the VX_MARS and VX_EV1 code bases. Please refer to the MARS Firmware Integration Manual for instructions to install these code bases and to program the onboard ATMEGA in the Arduino IDE.

- **MARS Firmware Integration Manual**
  https://www.vx-inc.com/technical-resources

- **VX_MARS**
  https://github.com/VX-inc/VX_MARS
  For use with the MARS system and the BlueShift.
- **VX_EV1**
  https://github.com/VX-inc/VX_EV1
  For use with the EV1 Development Board that implements the MARS system. Note, this codebase requires the VX_MARS library.

Application Examples

Initialize the MARS Library

Arduino

The following lines in the MARS application example initialize the MARS object. Note that mars objects can be renamed if desired and that all other mars function calls in the MARS example will need to be modified.

```
MARS mars;  //Create base AR system object, for adjusting display brightness, and reading ambient light sensor.
mars.init();  //Initialize the base hardware, display brightness and Ambient Light Sensor
```

User Buttons

There are 2 user buttons on the MARS EV1 Development Board and the BlueShift ODN-50005. The two implementations are slightly different in that, the MARS EV1 Dev. Board uses direct pin connections, and the BlueShift ODN-50005 uses I2C to read the button state.

Arduino - ODN-50005

To use the buttons on the ODN-50005, first, declare the ODN-50005 object.

```
BlueShift_ODN50005 blueshift_ODN50005; //Create object for ODN-50005 Imaging Sensor Module, contains buttons, and LEDs
```

Then initialize the object. note, you must include the Button Hold time, for holding the button can implement a different function if desired.

```
blueshift_ODN50005.init(BUTTON_HOLD_TIME_MS);
```

Now that the button manager is initialized, place the button manager function call in a polled loop, the button manager will report the button state, this is fed into a switch case with all of the potential button cases as defined by the button_state_t

```
 button_state_t buttonPressed = blueshift_ODN50005.buttonManager(); //Gets the state of the
button presses

  switch (buttonPressed) {
    case BUTTON_NO_PRESS:
      Break;
....
```

Here is the definition of button_state_t:

```
enum button_state_t {
       BUTTON_NO_PRESS,
       BUTTON_SHORT_UP,
       BUTTON_LONG_UP,
       BUTTON_SHORT_DOWN,
       BUTTON_LONG_DOWN,
       BUTTON_BOTH_PRESS
};
```

Arduino - EV1

The function calls for EV1 are identical to the ODN50005, except the name of the module is different (i.e. EV1.init()). Note that EV1 uses pins D5 and D6 for the buttons, the dip switches (SW3, SW4, SW5) on the EV1 board must all be set to 1 down, and 2 up to connect the buttons and ALS sensor.

Backlight Brightness Control of the CNED

The backlight brightness can be controlled by the ATMEGA to 10 adjustable brightness levels.

0 - Off
1 - 9 Intermediate Brightness Levels
10 - Maximum Brightness

The brightness is interpreted based on the timing of the ATMEGA output GPIO pin PB7.

Arduino

On the Arduino implementation, the brightness can be set utilizing the MARS library function calls as follows:

```
mars.setDisplayBrightness(display_brightness);
```

Where display_brightness is an integer value corresponding to the brightness level desired to be set. Once this value is set, the display will continue to operate at this value until set again. This value is driven by the PWM hardware, please check the ATMEL documentation if you intend to change any PWM Hardware settings, it may interfere with the operation of the brightness setting pin.

Lights

The ODN-50005 board controls 2 lights using PWM pins:

- White LED: PC6 (D5 Arduino)
- IR LED: PD7 (D6 Arduino)

These can be adjusted to any brightness using a 0-100% duty cycle PWM signal.

Arduino - ODN-50005

Ambient Light Sensor

The ambient light sensor is read as an analog input on the ATMEGA as pin PF7. This signal ranges from near 0V to near 5V depending on the ambient light levels.

Arduino

```
void ALSLoop(void) {
  static float average_brightness = display_brightness;
  if (als_controlled_brightness) {
    static uint8_t lastBrightnessValue = 0;
    int alsValue = mars.getALSValue(); //Get the value of the ambient light sensor, scale
from 0 to 1024 for 0-5V
    alsValue = map(alsValue, 0, 1024, 1, 10); //remaps the ambient light sensor output to a
1-10 range
    average_brightness = (0.99)*average_brightness + (0.01)*alsValue; //Adds a fade to the
ALS brightness adjustment
    display_brightness = (uint8_t) average_brightness;
    if (display_brightness != lastBrightnessValue) { //Sets the brightness only on a change
in value, only needed to not spam the serial monitor
      setBrightness(display_brightness);
    }
    lastBrightnessValue = display_brightness;
  }
}
```

USB Serial, for active communication over the USB-C connection

The microcontroller by default enumerates as a virtual comm port. This allows the user to open that comm port and define any communication protocol that is desired to implement the user's application. Keep in mind that the user should maintain a non-blocking style of programming to ensure that any polling loops (ALS or button presses) continue to function.

Important Notice – Please Read Carefully

***

VX Inc. reserves the right to make changes, modifications, and corrections to VX products and/or to this document at any time without notice. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

Customers are solely responsible for the use of VX products and VX assumes no liability for application assistance or the design of Customers' products.

Storage: 22°C at 50% relative humidity is recommended. Prolonged storage is not recommended.

Resale of VX products shall void any warranty granted by VX for such products.

This product shall not be used in life-support devices or other medical systems. Customer to independently verify information and shall test for all required certifications, including but not limited to, RoHS, ANSI Z87, and FDA.

***

Augmented Reality
Design
Displays
Integration