Introduction to ARM-GCC using Vaman

G V V Sharma*

Contents

Abstract—This document provides a simple introduction to ARM programming using Vaman.

1 Software

All codes used in this document are available at the following link

```
https://github.com/gadepall/vaman/
    tree/master/arm/codes/setup
```

2 Setup

2.1. Connect the raspberry pi to Vaman through USB.

2.2. On the left of the USB port, an LED and a button can be seen. Another button is visible on the right of the USB port.

2.3. Press the right button and immediately press the left button. The green LED starts blinking. The Vaman is now in download mode. mode.

2.4. Open termux on android and execute the following codes

```
cd ~
svn co https://github.com/
    gadepall/vaman/trunk/arm/
    codes/setup/blink
cd GCC_Project
make
scp  output/bin/blink.bin
    pi@192.168.0.114:
```

Appropriately modify the above ip address before sending blink.bin to the pi.

2.5. Now log on to the RPi and execute teh following

```
sudo python3 /home/pi/Vaman-dev
    /Vaman-sdk/TinyFPGA-
    Programmer-Application/
    tinyfpga-programmer-gui.py
    --port /dev/ttyACM0  --m4app
    blink.bin --mode m4-fpga
```

2.6. Enter the appropriate USB device port above while executing. Press the button to the right after the above command is successfully executed. The LED will start blinking.

3 Delay

3.1. See the following lines of the code below

```
codes/setup/blink/src/main.c
```

```
PyHal_Set_GPIO(18,1);//blue
PyHal_Set_GPIO(21,1);//
    green
PyHal_Set_GPIO(22,1);//red
    HAL_DelayUSec(2000000);
PyHal_Set_GPIO(18,0);
PyHal_Set_GPIO(21,0);
PyHal_Set_GPIO(22,0);
```

We may conclude that the blink delay is 2000 000us = 2 s.

3.2. Replace the following line in 3.1

```
HAL_DelayUSec(2000000);
```

with

```
HAL_DelayUSec(1000000);
```

and execute. Can you see any difference in the blink period?

3.3. To obtain red colour, execute the following code.

Type	Pin	Destination
Input	IO_5	GND

TABLE 3.5.1: Vaman control through external input.

```
codes/setup/red/src/main.c
```

Now obtain blue colour.

3.4. Now obtain green colour without blink.
Solution: Execute the following code.

```
codes/setup/onoff/src/main.c
```

3.5. Using Table 3.5.1 and Fig. 3.5.1, use an input pin to control the onboard LED. Solution: Execute the following code. You should see the LED blinking pink. Disconnecting the wire from GND will result in the LED blinking white and green alternately.
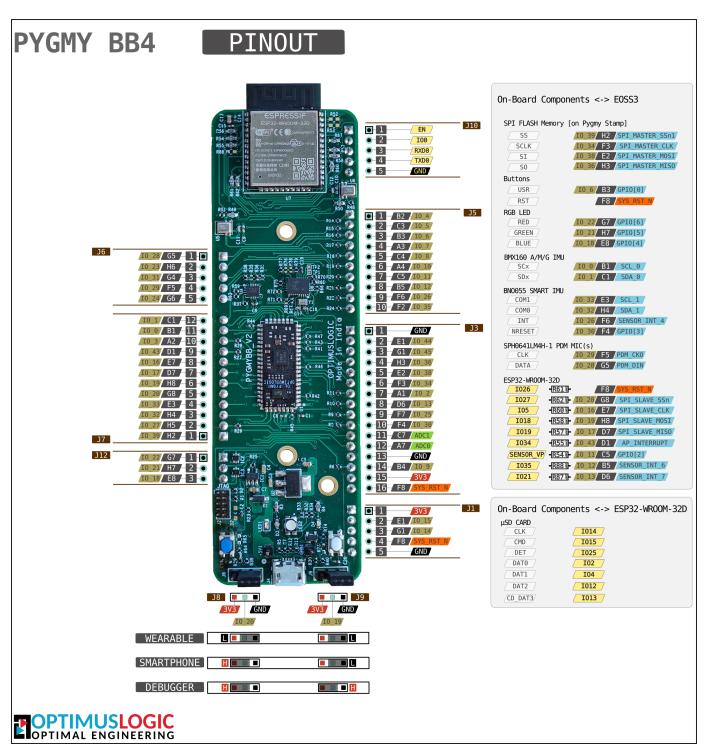
```
codes/setup/gpio/src/main.c
```

PYGMY BB4 PINOUT

J10
1	EN
2	IO0
3	RXD0
4	TXD0
5	GND

J5
1	B2	IO 4
2	C3	IO 5
3	B3	IO 6
4	A3	IO 7
5	C4	IO 8
6	A4	IO 10
7	C5	IO 11
8	B5	IO 12
9	F6	IO 26
10	F2	IO 35

J6
IO 28	G5	1
IO 23	H6	2
IO 31	G4	3
IO 29	F5	4
IO 24	G6	5

J7
IO 1	C1	12
IO 0	B1	11
IO 3	A2	10
IO 43	D1	9
IO 16	E7	8
IO 17	D7	7
IO 19	H8	6
IO 20	G8	5
IO 33	E3	4
IO 32	H4	3
IO 27	H5	2
IO 39	H2	1

J12
IO 22	G7	1
IO 21	H7	2
IO 18	E8	3

J3
1	GND
2	E1	IO 44
3	G1	IO 45
4	H3	IO 36
5	E2	IO 38
6	F3	IO 34
7	A1	IO 2
8	D6	IO 13
9	F7	IO 25
10	F4	IO 30
11	C7	ADC1
12	A7	ADC0
13	GND
14	B4	IO 9
15	3V3
16	F8	SYS_RST_N

J1
1	3V3
2	E1	IO 15
3	G1	IO 14
4	F8	SYS_RST_N
5	GND

J8 / **J9**
3V3 | GND | 3V3 | GND
IO 20 | IO 19

WEARABLE — L / L
SMARTPHONE — H / L
DEBUGGER — H / H

On-Board Components <-> EOSS3

SPI FLASH Memory [on Pygmy Stamp]
SS	IO 39	H2	SPI_MASTER_SSn1
SCLK	IO 34	F3	SPI_MASTER_CLK
SI	IO 38	E2	SPI_MASTER_MOSI
SO	IO 36	H3	SPI_MASTER_MISO

Buttons
| USR | IO 6 | B3 | GPIO[0] |
| RST | | F8 | SYS_RST_N |

RGB LED
RED	IO 22	G7	GPIO[6]
GREEN	IO 21	H7	GPIO[5]
BLUE	IO 18	E8	GPIO[4]

BMX160 A/M/G IMU
| SCx | IO 0 | B1 | SCL_0 |
| SDx | IO 1 | C1 | SDA_0 |

BNO055 SMART IMU
COM1	IO 33	E3	SCL_1
COM0	IO 32	H4	SDA_1
INT	IO 26	F6	SENSOR_INT_4
NRESET	IO 30	F4	GPIO[3]

SPH0641LM4H-1 PDM MIC(s)
| CLK | IO 29 | F5 | PDM_CKO |
| DATA | IO 28 | G5 | PDM_DIN |

ESP32-WROOM-32D
IO26	R61		F8	SYS_RST_N
IO27	R62	IO 20	G8	SPI_SLAVE_SSn
IO5	R60	IO 16	E7	SPI_SLAVE_CLK
IO18	R58	IO 19	H8	SPI_SLAVE_MOSI
IO19	R57	IO 17	D7	SPI_SLAVE_MISO
IO34	R55	IO 43	D1	AP_INTERRUPT
SENSOR_VP	R54	IO 11	C5	GPIO[2]
IO35	R88	IO 12	B5	SENSOR_INT_6
IO21	R87	IO 13	D6	SENSOR_INT_7

On-Board Components <-> ESP32-WROOM-32D

μSD CARD
CLK	IO14
CMD	IO15
DET	IO25
DAT0	IO2
DAT1	IO4
DAT2	IO12
CD_DAT3	IO13

OPTIMUSLOGIC — OPTIMAL ENGINEERING

Fig. 3.5.1: Pin Diagram