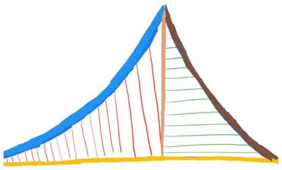
Embedd Systems Through Vaman



G. V. V. Sharma

First manual appeared in 2015

CONTENTS

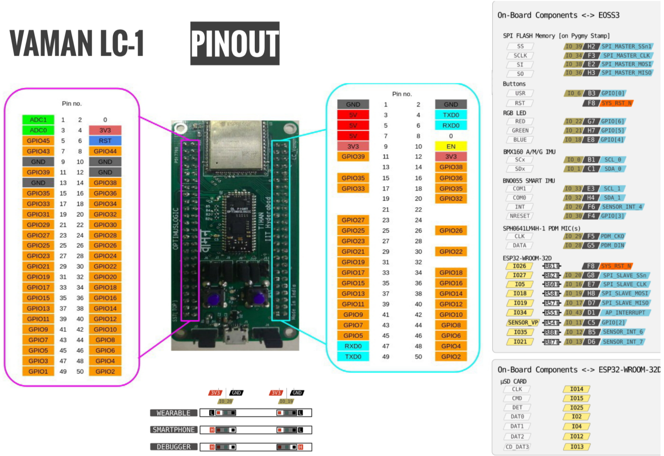


Fig. 1.1: Vaman pins. Right side reperesents ESP32 and left side M4-FPGA.

1 Blink

1.1 ESP

Here we show how to program the ESP32 on the Vaman using the Arduino framework.

1. Make sure that Vaman board does not power any devices.
2. Make connections as shown in Table 1.1.
3. The Vaman pin diagram is available in Figure 1.1.

VAMAN-ESP	UART PINS
5V	5V
GND	GND
TXD0	TXD
RXD0	RXD
0	GND

TABLE 1.1

4. Connect the Arduino-UART to raspberry pi through USB.
5. Connect the Vaman-ESP pins to the seven segment display according to Table 1.2

ESP	SEVEN SEGMENT DISPLAY
5V	COM
2	DOT

TABLE 1.2

6. On termux on your phone,

```
cd vaman/esp32/codes/ide/blink
pio run
```

7. Transfer the ini and bin files to the rpi

```
scp platformio.ini pi@192.168.50.252:./hi/platformio.ini

scp .pio/build/esp32doit−devkit−v1/firmware.bin
      pi@192.168.50.252:./hi/.pio/build/esp32doit−devkit
      −v1/firmware.bin
```

8. On rpi,

```
cd /home/pi/hi
pio run −t nobuild −t upload
```

9. On your phone, open

```
src/main.cpp
```

and change the delay to

```
delay(100);
```

and execute the code by following the steps above.
10. Flash the following code.

```
vaman/esp32/codes/ide/ota/setup
```

after entering your wifi username and password (in quotes below)

```
#define STASSID ”...” // Add your network credentials
#define STAPSK ”...”
```

in src/main.cpp file
11. You should be able to find the ip address of your vaman-esp using

```
ifconfig
nmap −sn 192.168.231.1/24
```

where your computer’s ip address is the output of ifconfig and given by 192.168.231.x
12. Assuming that the username is gvv and password is abcd, flash the following code wirelessly

```
vaman/esp32/codes/ide/ota/blink
```

through

```
pio run
pio run −t nobuild −t upload −−upload−port
      192.168.231.245
```

where you may replace the above ip address with the ip address of your vaman-esp.

13. Connect pin 2 to an LED to see it blinking.
14. Connect the pins between Vaman-ESP32 and Vaman-PYGMY as per Table 1.3

ESP32	**Vaman**
GPIO2	GPIO18
GPIO4	GPIO21
GPIO5	GPIO22

TABLE 1.3

15. Flash the following code OTA

```
vaman/esp32/codes/ide/ota/blinkt
```

You should see the onboard LEDs blinking.

16. Change the blink duration to 100 ms.

1.2 FPGA

We show how to program the Vaman FPGA/microcontroller board.

1. Follow the instructions available in the video at

```
https://github.com/whyakari/TermuxDisableProcces?tab=
      readme−ov−file
```

to ensure that termux is not killed during the following installation process.

2. On termux-debian,

```
wget https://raw.githubusercontent.com/gadepall/fwc−1/
      main/scripts/setup.sh
bash setup.sh
```

3. Login to termux-debian on the android device and execute the following commands

```
cd vaman/fpga/setup/codes/blink
source ˜/.vamenv/bin/activate
ql_symbiflow −compile −src vaman/fpga/setup/codes/
      blink −d ql−eos−s3 −P PU64 −v helloworldfpga.v
       −t helloworldfpga −p quickfeather.pcf −dump
      binary
scp blink/helloworldfpga.bin pi@192.168.0.114:
```

Make sure that the appropriate IP address for the raspberry pi is given in the above command.

4. Now execute the following commands on the raspberry pi.

```
python3 −m venv ˜/.vamenv
source ˜/.vamenv/bin/activate
git clone −−recursive https://github.com/QuickLogic−
      Corp/TinyFPGA−Programmer−Application.git
pip3 install tinyfpgab
deactivate
sudo reboot
source ˜/.vamenv/bin/activate
python3 TinyFPGA−Programmer−Application/tinyfpga
      −programmer−gui.py −−port /dev/ttyACM0 −−
      appfpga /home/pi/helloworldfpga.bin −−mode fpga
```

5. Make sure that the correct USB port address is given in the above command. Then press the button to the right of the USB port. After some time, the LED will start blinking red.
6. Replace the following line in the code in instruciton 8

```
assign redled = led; //If you want to change led colour
      to red,
```

with

```
assign blueled = led;
```

and execute the code.

7. Now modify the helloworldfpga.v file to get the green led blinking.
8. In the following verilog program,

```
codes/blink/helloworldfpga.v
```

pay attention to the following lines

```
delay = delay+1;
if(delay > 20000000)
begin
delay=25'b0;
led=!led;
end
```

It may be deduced from the above that the blink frequency is 20 MHz.

9. In instruction 8, replace

```
if(delay > 20000000)
```

with

```
if(delay==25'b1001100010010110100000000)
```

and execute the verilog code.

10. Since the delay is 20 MHz, the blink period is 1 second. Modify the verilog code so that the blink period becomes 0.5s.
11. Find the bit length of 20 MHz.
Solution:

$$\log_2(20000000) \approx 25 \tag{1.1}$$

12. Obtain the above answer using a Python code.
Solution: Exeucte the following code and compare with instruction 9.

```
codes/blink/freq_count.py
```

13. Ensure that the LED stays on in green colour.
Solution: Execute the following code

```
vaman/setup/codes/blink/onoff.v
```

14. Using Table 1.4 and Fig. **??**, control the onboard LED through an external input. Connect an external LED and control it using an output pin as well.

Type	Vaman Pin	Connection
Input	IO_28	GND
Output	IO_11	LED

TABLE 1.4: Vaman Input/Output.

Solution: Execute the following code and take out the input pin connect to GND. Plug it again. Do this repeatedly.

```
vaman/setup/codes/input/blink_ip.v
vaman/setup/codes/input/pygmy.pcf
```

1.3 ARM

We show how to control an LED using the M4 on Vaman.

1. Check your path

```
cd vaman/arm/setup/blink/GCC_Project
nvim config.mk
```

   and

2. and modify so that you have the following lines

```
#export PROJ_ROOT=/data/data/com.termux/files/home
      /pygmy−dev/pygmy−sdk
export PROJ_ROOT=/root/pygmy−dev/pygmy−sdk
```

3. Now execute

```
cd vaman/arm/setup/blink/GCC_Project
make −j4
scp output/bin/blink.bin pi@192.168.0.114:
```

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

4. Now log on to the RPi and execute the 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
```

5. 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.
6. 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.

7. Replace the following line in 6

Type	Pin	Destination
Input	IO_5	GND

TABLE 1.5: Vaman control through external input.

```
        HAL_DelayUSec(2000000);
```

   with

```
        HAL_DelayUSec(1000000);
```

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

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

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

   Now obtain blue colour.

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

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

10. Using Table 1.5 and Fig. **??**, 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.

```
vaman/arm/codes/setup/gpio/src/main.c
```