

*****Draft*****

learning Circle Bare Metal
Testing on RPi Zero
05/06/22

*****Draft*****

Testing the libpico.a from circle for a RPi Zero.

```
cd addon/pico/  
make clean  
  CLEAN /home/devel/circle/addon/pico  
make  
  CPP swdloader.o  
  AR libpico.a  
  
cp libpico.a ../../my-doc/  
  
arm-none-eabi-objdump -d libpico.a > libpico.a.txt
```

Following the posting on the Ultibo project

<https://ultibo.org/forum/viewtopic.php?f=9&t=1640>

1.0 Ultibo could provide the same functions as RPi4B running OpenOCD with Single Wire Debugger SWD to program a Pico using 3 GPIO pins.
This would require compiling the openocd C for use for linking with Ultibo.
I recently tested an updated version of openocd on 05/03/22

https://github.com/develone/Ultibo_Proj...-build.pdf

this updated version worked as previous version.

2.0 Using a Pico running picoprobe and Ultibo provide the functions of OpenOCD.

<https://www.youtube.com/watch?v=jnC5LrTx470&t=147s>

<https://www.digikey.com/en/maker/project/...6f67f184c0>

<https://github.com/raspberrypi/picoprobe...master/src>

<https://github.com/develone/picoprobe>

3.0 Running OpenOCD in Linux Raspberry Pi O/S and using MicroDos by pik33

Ultibo posted

It might be much simpler than compiling OpenOCD, we noticed recently that our friends at the [Circle](#) project added [SWD loader](#) support for the Pico.

This uses just the GPIO pins and looks like it would be fairly minor to implement in Ultibo as well.

Initial tests following forking the <https://github.com/rsta2/circle>

<https://github.com/rsta2/circle/tree/master/addon/pico>

README

This sample loads and starts a program into the Raspberry Pi Pico, which has to be connected as follows (SoC numbers):

Raspberry Pi Pico	Raspberry Pi 1-4 or Zero
-------------------	--------------------------

SWCLK	<---	GPI017
SWDIO	<--->	GPI027
RUN	<---	GPI024
GND	<--->	GND

This is the default setting. If you want to different GPIO pins, you have to update the configuration in the file kernel.cpp.

The program loads the file firmware.bin to the Raspberry Pi Pico, which does simply blink the LED. If you want to load your own program, you have to replace this file. Be sure to copy the .bin file from the built Pico SDK project and rename it to firmware.bin in this directory. Your Pico project must be configured for operation from RAM using this option in CMakeLists.txt, where PROJECT is the project name:

```
pico_set_binary_type (PROJECT no_flash)
```

You should install the file dummy_flash.uf2 in the flash of your Raspberry Pi Pico before running the program.

```
git clone git@github.com:develone/circle.git
```

```
cd circle
```

```
./makeall
```

```
TOOL converttool
CPP  actled.o
CPP  alloc.o
CPP  assert.o
CPP  bcmframebuffer.o
CPP  bcmmailbox.o
CPP  bcmpropertytags.o
CPP  bcmwatchdog.o
CPP  chargenerator.o
CPP  classallocator.o
CPP  cputhrottle.o
CPP  debug.o
AS   delayloop.o
CPP  device.o
CPP  devicenameservice.o
CPP  dmachannel.o
CPP  dmasoundbuffers.o
CPP  gpiloclock.o
```

CPP gpiomanager.o
CPP gpiopin.o
CPP gpiopinfiq.o
CPP i2cmaster.o
CPP i2cslave.o
CPP hdmioundbaseddevice.o
CPP i2ssoundbaseddevice.o
CPP koptions.o
CPP logger.o
CPP machineinfo.o
CPP multicore.o
CPP nulldevice.o
CPP ptrarray.o
CPP ptrlist.o
CPP pwmoutput.o
CPP pwmsoundbaseddevice.o
CPP pwmsounddevice.o
CPP qemu.o
CPP screen.o
CPP serial.o
CPP soundbaseddevice.o
CPP spimaster.o
CPP spimasteraux.o
CPP spimasterdma.o
CPP spinlock.o
CPP string.o
CPP sysinit.o
CPP time.o
CPP timer.o
CPP tracer.o
CPP usertimer.o
CPP util.o
AS util_fast.o
CPP virtualgpiopin.o
CPP chainboot.o
CPP macaddress.o
CPP netdevice.o
CPP new.o
CPP heapallocator.o
CPP pageallocator.o
AS setjmp.o
CPP numberpool.o
CPP latencytester.o
CPP writebuffer.o
CPP 2dgraphics.o
CPP smimaster.o
AS cache-v7.o
CPP exceptionhandler.o
AS exceptionstub.o
CPP memory.o
CPP pagetable.o
AS startup.o

CPP synchronize.o
CPP bcmrandom.o
CPP interrupt.o
CPP purecall.o
CPP cxa_guard.o
AR libcircle.a
CPP lan7800.o
CPP smsc951x.o
CPP usbbluetooth.o
CPP usbcdcethernet.o
CPP usbconfigparser.o
CPP usbdevice.o
CPP usbdevicefactory.o
CPP usbendpoint.o
CPP usbfunction.o
CPP usbgamepad.o
CPP usbgamepadps3.o
CPP usbgamepadps4.o
CPP usbgamepadstandard.o
CPP usbgamepadswitchpro.o
CPP usbgamepadxbox360.o
CPP usbgamepadxboxone.o
CPP usbhiddevice.o
CPP usbhostcontroller.o
CPP usbkeyboard.o
CPP usbmassdevice.o
CPP usbmidi.o
CPP usbmouse.o
CPP usbprinter.o
CPP usbrequest.o
CPP usbstandardhub.o
CPP usbstring.o
CPP usbserial.o
CPP usbserialch341.o
CPP usbserialcp2102.o
CPP usbserialpl2303.o
CPP usbserialft231x.o
CPP usbserialcdc.o
CPP usbtouchscreen.o
CPP dwhcidevice.o
CPP dwhciframeschednper.o
CPP dwhciframeschednsplit.o
CPP dwhciframeschedper.o
CPP dwhciregister.o
CPP dwhcirootport.o
CPP dwhcixactqueue.o
CPP dwhcixferstagedata.o
AR libusb.a
CPP keyboardbehaviour.o
CPP keymap.o
CPP mousebehaviour.o
CPP mouse.o

CPP touchscreen.o
CPP rpitouchscreen.o
CPP console.o
CPP keyboardbuffer.o
CPP linediscipline.o
AR libinput.a
CPP partition.o
CPP partitionmanager.o
AR libfs.a
CPP fatfs.o
CPP fatcache.o
CPP fatinfo.o
CPP fat.o
CPP fatdir.o
AR libfatfs.a
CPP task.o
CPP scheduler.o
AS taskswitch.o
CPP synchronizationevent.o
CPP mutex.o
CPP semaphore.o
AR libsched.a
CPP netsubsystem.o
CPP nettask.o
CPP netsocket.o
CPP socket.o
CPP transportlayer.o
CPP networklayer.o
CPP linklayer.o
CPP netdevlayer.o
CPP phytask.o
CPP arphandler.o
CPP icmphandler.o
CPP routecache.o
CPP netconnection.o
CPP udpconnection.o
CPP tcpconnection.o
CPP retransmissionqueue.o
CPP retranstimeoutcalc.o
CPP tcprejector.o
CPP netconfig.o
CPP ipaddress.o
CPP netqueue.o
CPP checksumcalculator.o
CPP dnsclient.o
CPP ntpclient.o
CPP mqttclient.o
CPP mqttsendpacket.o
CPP mqtreceivepacket.o
CPP dhcpclient.o
CPP ntpdaemon.o
CPP httpdaemon.o

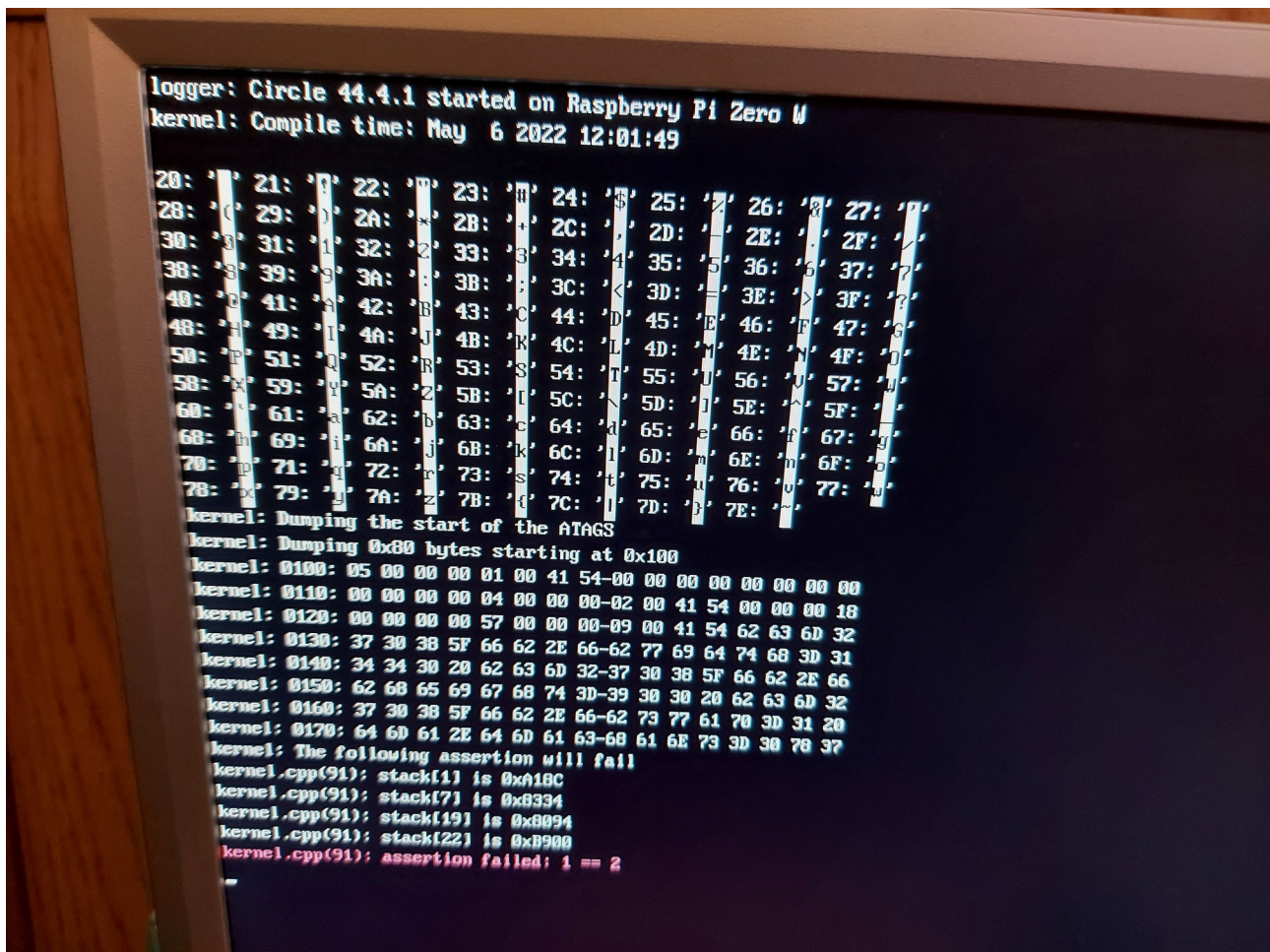
```

CPP httpclient.o
CPP tftpdaemon.o
CPP syslogdaemon.o
AR libnet.a
CPP main.o
CPP kernel.o
LD kernel.elf
DUMP kernel.lst
COPY kernel.img
WC kernel.img => 85088

```

cd ../

cd sample/03-screentext/



make

```

CPP main.o
CPP kernel.o
LD kernel.elf
DUMP kernel.lst
COPY kernel.img
WC kernel.img => 61028

```

cd ../17-fractal/


```
make
CPP main.o
CPP kernel.o
CPP mandelbrot.o
LD kernel.elf
DUMP kernel.lst
COPY kernel.img
WC kernel.img => 61252
```



update

