pictools Documentation

Release 0.17.0

Erik Moqvist

Sep 26, 2018

Contents

PIC tools is a collection of tools to ease PIC software development. PIC is family of microcontrollers made by Microchip.

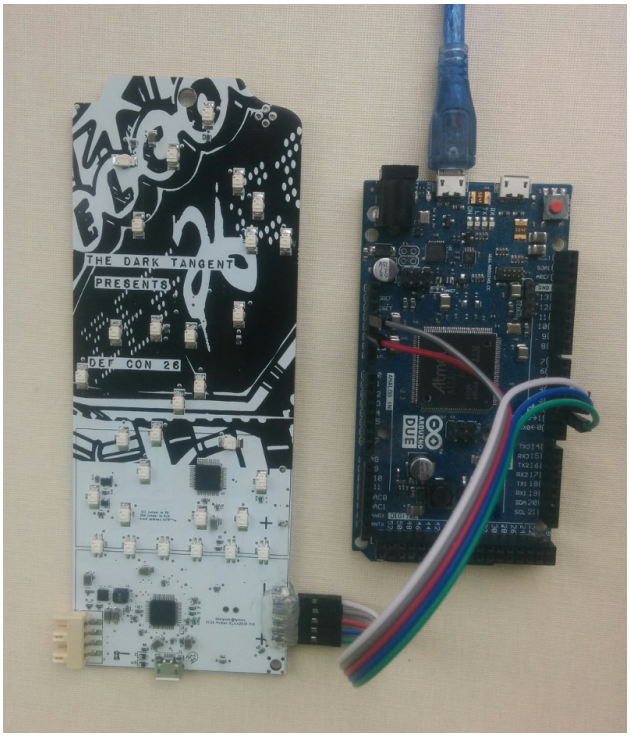
Features:

- A PIC programmer based on an Arduino Due. Today only the PIC32MM0256GPM064 family is supported.

Project homepage: https://github.com/eerimoq/pictools

CHAPTER 1

Hardware setup



The programmer (to the right) connected to a PC with serial over USB. The DEF CON 26 Badge (to the left) with a PIC32MM MCU to be programmed.

Note: The native USB port on the Arduino Due is used when programming the PIC, not the programming port as in the picture. The programming port is only used when programming the programmer.

Signal	Color	Arduino pin
MCLRN	white	D4
VDD (3V3)	grey	3V3
VSS (GND)	purple	GND
PGED	blue	D3
PGEC	green	D2

CHAPTER 2

Installation

1. Install pictools.

```
pip install pictools
```

2. Connect the Arduino Due programming USB port to the PC.
3. Upload the programmer application to the Arduino Due.

   Ubuntu:

   (a) Install bossac.

   ```
   > sudo apt install bossa-cli
   ```

   (b) Upload the programmer binary to the Arduino Due. Change serial port to match your setup.

   ```
   > pictools --port /dev/arduino programmer_upload
   Uploading programmer application version 0.10.0.
   Erase flash
   Write 23504 bytes to flash
   [==============================] 100% (92/92 pages)
   Set boot flash true
   CPU reset.
   Upload complete.
   ```

   Windows:

   (a) Install the Arduino IDE.

   (b) Start the Arduino IDE and click “Tools” -> “Board:” -> “Boards Manager…”.

   (c) Install *Arduino SAM Boards*.

   (d) Open a Windows PowerShell and upload the programmer binary to the Arduino Due. Change serial port and bossac path to match your setup.

```
> pictools --port COM5 programmer_upload --bossac-path
↪'c:\Users\erik\Documents\ArduinoData\packages\arduino\tools\bossac\1.6.1-
↪arduino'
Uploading programmer application version 0.10.0.
Erase flash
Write 23504 bytes to flash
[==============================] 100% (92/92 pages)
Set boot flash true
CPU reset.
Upload complete.
```

If necessary, give `-u` to the upload command above to unlock any locked flash regions.

4. Connect the Arduino Due native USB port to the PC.
5. Reset the Arduino Due by pressing its reset button.
6. Done!

CHAPTER 3

Command line tool

Descriptions and example usages of the most commonly used subcommands in the command line tool `pictools`.

Add the `--mcu` option before the subcommand to select your MCU.

3.1 Write to flash

Write given file `hello_world.s19` to flash. Optionally performs erase and read back verify operations.

```
> pictools --port /dev/arduino flash_write --chip-erase hello_world.s19
Programmer is alive.
PIC reset.
Erasing the chip.
Chip erase complete.
Connected to PIC.
Writing /home/erik/workspace/pictools/hello_world.s19 to flash.
100%|| 12052/12052 [00:00<00:00, 65081.89 bytes/s]
Write complete.
```

3.2 Read from flash

Read from the flash memory.

```
> pictools --port /dev/arduino flash_read 0x1d000000 0x1000 memory.s19
Programmer is alive.
PIC is alive.
Reading 0x1d000000-0x1d001000.
100%|| 4096/4096 [00:00<00:00, 60882.44 bytes/s]
Read complete.
```

3.3 Read the whole flash

Read program flash, boot flash and configuration memory.

```
> pictools --port /dev/arduino flash_read_all memory.s19
Programmer is alive.
PIC is alive.
Reading 0x1d000000-0x1d040000.
100%|| 262144/262144 [00:04<00:00, 61164.56 bytes/s]
Read complete.
Reading 0x1fc00000-0x1fc01800.
100%|| 6144/6144 [00:00<00:00, 59445.91 bytes/s]
Read complete.
```

3.4 Erase a flash range

Erase given flash range.

```
> pictools --port /dev/arduino flash_erase 0x1d000000 0x1000
Programmer is alive.
PIC is alive.
Erasing 0x1d000000-0x1d001000.
Erase complete.
```

3.5 Chip erase

Erases program flash, boot flash and configuration memory.

```
> pictools --port /dev/arduino flash_erase_chip
Erasing the chip.
Programmer is alive.
Chip erase complete.
```

3.6 Reset

Reset the PIC.

```
> pictools --port /dev/arduino reset
Programmer is alive.
PIC reset.
```

3.7 Print the configuration memory

Print the configuration memory.

```
> pictools --port /dev/arduino configuration_print
Programmer is alive.
PIC is alive.
FDEVOPT
  USERID: 65535
  FVBUSIO: 0
  FUSBIDIO: 1
  ALTI2C: 1
  SOSCHP: 1
FICD
  ICS: 2
  JTAGEN: 0
FPOR
  LPBOREN: 1
  RETVR: 1
  BOREN: 3
FWDT
  FWDTEN: 0
  RCLKSEL: 3
  RWDTPS: 20
  WINDIS: 1
  FWDTWINSZ: 3
  SWDTPS: 20
FOSCSEL
  FCKSM: 1
  SOSCSEL: 0
  OSCIOFNC: 1
  POSCMOD: 3
  IESO: 1
  SOSCEN: 0
  PLLSRC: 1
  FNOSC: 0
FSEC
  CP: 1
```

3.8 Print the device id

Print the device id.

```
> pictools --port /dev/arduino device_id_print
Programmer is alive.
PIC is alive.
DEVID
  VER: 2
  DEVID: 0x0773c053
```

3.9 Print the UDID

Print the unique chip id.

```
> pictools --port /dev/arduino udid_print
Programmer is alive.
```

(continues on next page)

(continued from previous page)

```
PIC is alive.
UDID
  UDID1: 0xff918406
  UDID2: 0xff524000
  UDID3: 0xffffff25
  UDID4: 0xffff0219
  UDID5: 0xffff0280
```

3.10 Ping the programmer

Test if the programmer is alive.

```
> pictools --port /dev/arduino programmer_ping
Programmer is alive.
```

3.11 Read the programmer software version

Read the programmer software version from the programmer.

```
> pictools --port /dev/arduino programmer_version
Programmer is alive.
0.10.0
```

3.12 Ping the PIC

Test if the PIC is alive and executing the RAM application.

```
> pictools --port /dev/arduino ping
Programmer is alive.
PIC is alive.
```

CHAPTER 4

Test matrix

A list of all supported MCUs and their test results.

Write time is the time it takes to connect to the PIC, perform a chip erase and write zeros to the whole memory. An example measurement for PIC32MM0256GPM064 can be seen below.

```
> time pictools -p /dev/arduino -m pic32mm0256gpm064 flash_write -c zeros.s19
Programmer is alive.
PIC reset.
Erasing the chip.
Chip erase complete.
Connected to PIC.
Writing /home/erik/workspace/pictools/zeros.s19 to flash.
100%|| 268288/268288 [00:03<00:00, 67785.34 bytes/s]
Write complete.

real    0m4.565s
user    0m0.453s
sys     0m0.046s
```

MCU	Memory size	Write time	Comment
PIC32MM0064GPM028	70k	Not tested	
PIC32MM0128GPM028	134k	Not tested	
PIC32MM0256GPM028	262k	Not tested	
PIC32MM0064GPM036	70k	Not tested	
PIC32MM0128GPM036	134k	Not tested	
PIC32MM0256GPM036	262k	Not tested	
PIC32MM0064GPM048	70k	Not tested	
PIC32MM0128GPM048	134k	Not tested	
PIC32MM0256GPM048	262k	Not tested	
PIC32MM0064GPM064	70k	Not tested	
PIC32MM0128GPM064	134k	Not tested	
PIC32MM0256GPM064	262k	4.6 s	

CHAPTER 5

Similar projects

There are a bunch of projects similar to *PIC tools*. Here are a few of them:

- https://github.com/WallaceIT/picberry
- https://github.com/sergev/pic32prog
- http://usbpicprog.org/
- https://wiki.kewl.org/dokuwiki/projects:pickle
- http://picpgm.picprojects.net/
- https://github.com/jaromir-sukuba/a-p-prog
- https://github.com/G4me4u/pic-programmer-arduino
- https://github.com/rweather/ardpicprog