pyavrutils Documentation

Release 0.0.1

ponty

August 22, 2011

CONTENTS

pyavrutils

Date August 22, 2011

PDF pyavrutils.pdf

Contents:

pyavrutils can build AVR and arduino code from python

Links:

- home: https://github.com/ponty/pyavrutils
- documentation: http://ponty.github.com/pyavrutils

Features:

- python wrapper for avr-gcc, avr-size, arscons
- build files or strings (strings are saved as temp files)
- MCU list
- get code size using avr-size
- avr-gcc default is optimized for size

Known problems:

- Python 3 is not supported
- temp files are not removed
- **arscons has some problems:**
  - it builds bigger programs
  - compile error in some cases

Possible usage:

- experimenting with flags
- building from paver
- unit tests
- building arduino code without GUI

CHAPTER

ONE

BASIC USAGE

```
>>> from pyavrutils import AvrGcc
>>> cc = AvrGcc()
>>> cc.build('int main(){}')
>>> cc.size().program_bytes
66

>>> from pyavrutils import Arduino
>>> cc = Arduino()
>>> cc.mcu = 'atmega8'
>>> cc.build('void setup(){};void loop(){}')
>>> cc.size().program_bytes
1612
```

CHAPTER

TWO

INSTALLATION

2.1 General

- arscons is already included in the library
- install setuptools
- install gcc-avr
- install scons (only for arscons)
- install arduino (only for arscons)
- install the program:

if you have setuptools installed:

```
# as root
easy_install pyavrutils
```

2.2 Ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install binutils-avr
sudo apt-get install gcc-avr
sudo apt-get install scons
sudo apt-get install arduino
sudo easy_install pyavrutils
```

2.3 Uninstall

using pip:

```
# as root
pip uninstall pyavrutils
```

CHAPTER

THREE

USAGE

3.1 AVR

```
>>> from pyavrutils import AvrGcc
>>> cc = AvrGcc(mcu='atmega48')
>>> cc.targets
['at43usb320', 'at43usb355', 'at76c711', 'at86rf401', 'at90c8534', 'at90can128', 'at90can32', 'at90ca
>>> cc.options_generated()
['avr-gcc', '-Df_cpu=4000000', '-mmcu=atmega48', '--std=gnu99', '-Wl,--relax', '-Wl,--gc-sections', '
>>> cc.build('int main(){}')
>>> cc.output
'/tmp/pyavrutils_Hxi5MA.elf'
>>> cc.size()
AvrSize <prog:80 bytes 2.0% mem:0 bytes 0.0% >
>>> cc.size().program_bytes
80
>>> cc.mcu='atmega168'
>>> cc.options_generated()
['avr-gcc', '-Df_cpu=4000000', '-mmcu=atmega168', '--std=gnu99', '-Wl,--relax', '-Wl,--gc-sections',
>>> cc.build('int main(){}')
>>> cc.output
'/tmp/pyavrutils_Hxi5MA.elf'
>>> cc.size().program_bytes
132
```

3.2 arduino

```
>>> from pyavrutils import Arduino
>>> cc = Arduino(board='mini')
>>> cc.build('void setup(){};void loop(){}')
>>> cc.output
path('/tmp/pyavrutils_8xLXlE/pyavrutils_H6igFd/pyavrutils_H6igFd.elf')
>>> cc.size()
AvrSize <prog:1802 bytes 11.0% mem:191 bytes 18.7% >
>>> cc.size().program_bytes
1802
>>> cc.board='pro'
>>> cc.build('void setup(){};void loop(){}')
>>> cc.output
path('/tmp/pyavrutils_dP1T5S/pyavrutils_SOu4nV/pyavrutils_SOu4nV.elf')
```

```
>>> cc.size().program_bytes
1826
```

CHAPTER

FOUR

EXAMPLES

4.1 Simple example

Example program:

```
'''
test minimum program size with different optimizations
'''

from pyavrutils import AvrGcc
from entrypoint2 import entrypoint

cc = AvrGcc()
code = 'int main(){}'

def test():
    print '    compiler option:', ' '.join(cc.options_generated())
    cc.build(code)
    print '    program size =', cc.size().program_bytes

@entrypoint
def main():
    print 'compiler version:', cc.version()
    print 'code:', code
    print
    print 'no optimizations::'
    print
    cc.optimize_no()
    test()
    print
    print 'optimize for size::'
    print
    cc.optimize_for_size()
    test()
```

Output:

```
$ python -m pyavrutils.examples.simple
compiler version: 4.3.5
code: int main(){}

no optimizations::

    compiler option: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99
```

```
    program size = 150

optimize for size::

    compiler option: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99 -Wl,--relax -Wl,--gc-sectio
    program size = 132
```

4.2 Test size with unused code

Example program:

```
from pyavrutils.avrgcc import AvrGcc
from entrypoint2 import entrypoint

cc = AvrGcc()

def test_option(sources, optimization, gc_sections=0, ffunction_sections=0):
    print 'optimization =', optimization,
    print 'gc_sections =', gc_sections,
    print 'ffunction_sections =', ffunction_sections,
    print

    cc.optimization = optimization
    cc.gc_sections = gc_sections
    cc.ffunction_sections = ffunction_sections
    try:
        cc.build(sources)
        size = cc.size()
        print 'program, data =', str(size.program_bytes).rjust(8) , ',', str(size.data_bytes).rjust(8
    except:
        print  'compile error'

def test(sources):
    print 'sources:', sources
    test_option(sources, 0)
    test_option(sources, 's',0)
    test_option(sources, 's',1)
    test_option(sources, 's',1,1)

@entrypoint
def main():
    cc.optimize_no()
    print  'compiler version:', cc.version()
    print  'compiler options:', ' '.join(cc.options_generated())
    print
    print 'minimum size'
    print 20 * '='
    test(['int main(){}'])

    print
    print 'unused function in separate file'
    print 40 * '='
    test(['int main(){}', 'int f(){return 2;}'])

    print
    print 'unused function in the same file'
```

```
    print 40 * '='
    test(['int main(){}; int f(){return 2;}'])
```

Output:

```
$ python -m pyavrutils.examples.deadcode
compiler version: 4.3.5
compiler options: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99

minimum size
====================
sources: ['int main(){}']
optimization = 0 gc_sections = 0 ffunction_sections = 0
program, data =      150 ,        0
optimization = s gc_sections = 0 ffunction_sections = 0
program, data =      138 ,        0
optimization = s gc_sections = 1 ffunction_sections = 0
program, data =      138 ,        0
optimization = s gc_sections = 1 ffunction_sections = 1
program, data =      138 ,        0

unused function in separate file
========================================
sources: ['int main(){}', 'int f(){return 2;}']
optimization = 0 gc_sections = 0 ffunction_sections = 0
program, data =      168 ,        0
optimization = s gc_sections = 0 ffunction_sections = 0
program, data =      144 ,        0
optimization = s gc_sections = 1 ffunction_sections = 0
program, data =      138 ,        0
optimization = s gc_sections = 1 ffunction_sections = 1
program, data =      138 ,        0

unused function in the same file
========================================
sources: ['int main(){}; int f(){return 2;}']
optimization = 0 gc_sections = 0 ffunction_sections = 0
program, data =      168 ,        0
optimization = s gc_sections = 0 ffunction_sections = 0
program, data =      144 ,        0
optimization = s gc_sections = 1 ffunction_sections = 0
program, data =      144 ,        0
optimization = s gc_sections = 1 ffunction_sections = 1
program, data =      138 ,        0
```

Conclusions:

- both `gc_sections` and `ffunction_sections` should be used

4.3 Test size with delay.h

Example program:

```
from pyavrutils.avrgcc import AvrGcc
from entrypoint2 import entrypoint

templ = '''
```

```
#include <avr/io.h>
#include <util/delay.h>
int main()
{
    %s;
    return 0;
}
'''

cc = AvrGcc()
cc.optimize_no()
print  'compiler version:', cc.version()
print

def test(snippet, option=''):
    print  snippet.ljust(33) ,
    cc.options_extra = option.split()
    print  'compiler option:', option, '\t',
    try:
        cc.build([templ % snippet])
        size = cc.size()
        print 'program, data =', str(size.program_bytes).rjust(8) , ',', str(size.data_bytes).rjust(8
    except:
        print  'compile error'

@entrypoint
def main():
    cc.optimization = 0

    test('_delay_ms(4)', '-O0')
    test('_delay_ms(4)', '-O1')
    test('_delay_ms(4)', '-O2')
    test('_delay_ms(4)', '-O3')
    test('_delay_ms(4)', '-Os')

    test('volatile double x=3;_delay_ms(x)', '-Os')
```

Output:

```
$ python -m pyavrutils.examples.delaysize
compiler version: 4.3.5

_delay_ms(4)                      compiler option: -O0        program, data =     3282 ,        8
_delay_ms(4)                      compiler option: -O1        program, data =      146 ,        0
_delay_ms(4)                      compiler option: -O2        program, data =      146 ,        0
_delay_ms(4)                      compiler option: -O3        program, data =      146 ,        0
_delay_ms(4)                      compiler option: -Os        program, data =      146 ,        0
volatile double x=3;_delay_ms(x)  compiler option: -Os        program, data =     3218 ,        8
```

Conclusions:

- parameter should be constant
- optimization should be 1, 2, 3 or s

4.4 Test size with program space

Example program:

```
from pyavrutils.avrgcc import AvrGcc
from entrypoint2 import entrypoint

templ = '''
#include <avr/io.h>
#include <avr/pgmspace.h>
int main()
{
    %s;
    return 0;
}
'''

cc = AvrGcc()
cc.optimization=0
print  'compiler version:', cc.version()
print  'compiler options:', ' '.join(cc.options_generated())
print

def test(snippet):
    print  snippet ,'\t\t',
    try:
        cc.build([templ % snippet])
        size = cc.size()
        print 'program, data =', str(size.program_bytes).rjust(8) , ',', str(size.data_bytes).rjust(8
    except:
        print  'compile error'


def test_comb(s):
    words='static const PROGMEM'.split()
    def choice(i):
        return [words[i],' '*len(words[i])]

    for s0 in choice(0):
        for s1 in choice(1):
            for s2 in choice(2):
#                   for s3 in choice(3):
                    test('%s %s char s[] %s = "%s"' % (s0,s1,s2,s))
@entrypoint
def main():
    test_comb("12345")
    test_comb("1234512345")
```

Output:

```
$ python -m pyavrutils.examples.pgmspace
compiler version: 4.3.5
compiler options: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99 -Wl,--relax -Wl,--gc-sections -

static const char s[] PROGMEM = "12345"                 program, data =      144 ,        0
static const char s[]         = "12345"                 program, data =      166 ,        0
static       char s[] PROGMEM = "12345"                 program, data =      144 ,        0
static       char s[]         = "12345"                 program, data =      166 ,        0
```

```
       const char s[] PROGMEM = "12345"                program, data =      260 ,         6
       const char s[]         = "12345"                program, data =      260 ,         6
             char s[] PROGMEM = "12345"                program, data =      260 ,         6
             char s[]         = "12345"                program, data =      260 ,         6
static const char s[] PROGMEM = "1234512345"               program, data =      144 ,         0
static const char s[]         = "1234512345"               program, data =      166 ,         0
static       char s[] PROGMEM = "1234512345"               program, data =      144 ,         0
static       char s[]         = "1234512345"               program, data =      166 ,         0
       const char s[] PROGMEM = "1234512345"               program, data =      266 ,        12
       const char s[]         = "1234512345"               program, data =      266 ,        12
             char s[] PROGMEM = "1234512345"               program, data =      266 ,        12
             char s[]         = "1234512345"               program, data =      266 ,        12
```

Conclusions:

- constant string should be static or global
- `const` has no effect on size
- PROGMEM should be used

4.5 Test minimum size

Example program:

```
'''
test minimum program size with all MCUs
'''

from entrypoint2 import entrypoint
from pyavrutils.avrgcc import AvrGcc, AvrGccCompileError

def test(cc, mcu):
    print 'MCU =', mcu.ljust(20),
    cc.mcu = mcu
    try:
        cc.build(cc.minprog)
        print '    program/data size =', cc.size().program_bytes, ',', cc.size().data_bytes
    except AvrGccCompileError:
        print '    compile error:', cc.error_text.splitlines()[0]

@entrypoint
def main():
    cc = AvrGcc()
    print '--------------'
    print 'avr-gcc'
    print '--------------'

    print 'compiler version:', cc.version()
    cc.optimize_for_size()
    print 'compiler options:', ' '.join(cc.options_generated())
    print 'code:', cc.minprog
    print
    for mcu in cc.targets:
        test(cc, mcu)
```

Output:

```
$ python -m pyavrutils.examples.minsize
--------------
avr-gcc
--------------
compiler version: 4.3.5
compiler options: avr-gcc -Df_cpu=4000000 -mmcu=atmega168 --std=gnu99 -Wl,--relax -Wl,--gc-sections -
code: int main(){};

MCU = at43usb320              program/data size = 80 , 0
MCU = at43usb355              program/data size = 80 , 0
MCU = at76c711                program/data size = 88 , 0
MCU = at86rf401               program/data size = 40 , 0
MCU = at90c8534               program/data size = 42 , 0
MCU = at90can128              program/data size = 202 , 0
MCU = at90can32               program/data size = 176 , 0
MCU = at90can64               program/data size = 176 , 0
MCU = at90pwm1                program/data size = 92 , 0
MCU = at90pwm2                program/data size = 92 , 0
MCU = at90pwm216              program/data size = 156 , 0
MCU = at90pwm2b               program/data size = 92 , 0
MCU = at90pwm3                program/data size = 92 , 0
MCU = at90pwm316              program/data size = 156 , 0
MCU = at90pwm3b               program/data size = 92 , 0
MCU = at90pwm81               program/data size = 68 , 0
MCU = at90s1200               compile error: /tmp/pyavrutils_QSd3Tx/pyavrutils_8bUb7H.c:1: error: MC
MCU = at90s2313               program/data size = 46 , 0
MCU = at90s2323               program/data size = 30 , 0
MCU = at90s2333               program/data size = 52 , 0
MCU = at90s2343               program/data size = 30 , 0
MCU = at90s4414               program/data size = 54 , 0
MCU = at90s4433               program/data size = 52 , 0
MCU = at90s4434               program/data size = 62 , 0
MCU = at90s8515               program/data size = 54 , 0
MCU = at90s8535               program/data size = 62 , 0
MCU = at90scr100              program/data size = 180 , 0
MCU = at90usb1286             program/data size = 206 , 0
MCU = at90usb1287             program/data size = 206 , 0
MCU = at90usb162              program/data size = 144 , 0
MCU = at90usb646              program/data size = 180 , 0
MCU = at90usb647              program/data size = 180 , 0
MCU = at90usb82               program/data size = 144 , 0
MCU = at94k                   program/data size = 172 , 0
MCU = ata6289                 program/data size = 82 , 0
MCU = atmega103               program/data size = 156 , 0
MCU = atmega128               program/data size = 194 , 0
MCU = atmega1280              program/data size = 282 , 0
MCU = atmega1281              program/data size = 258 , 0
MCU = atmega1284p             program/data size = 194 , 0
MCU = atmega128rfa1           program/data size = 342 , 0
MCU = atmega16                program/data size = 112 , 0
MCU = atmega161               program/data size = 112 , 0
MCU = atmega162               program/data size = 140 , 0
MCU = atmega163               program/data size = 100 , 0
MCU = atmega164a              program/data size = 152 , 0
MCU = atmega164p              program/data size = 152 , 0
MCU = atmega165               program/data size = 116 , 0
MCU = atmega165a              compile error: Known MCU names:
MCU = atmega165p              program/data size = 116 , 0
```

```
MCU = atmega168              program/data size = 132 , 0
MCU = atmega168a             program/data size = 80 , 0
MCU = atmega168p             program/data size = 132 , 0
MCU = atmega169              program/data size = 120 , 0
MCU = atmega169a             program/data size = 120 , 0
MCU = atmega169p             program/data size = 120 , 0
MCU = atmega169pa            program/data size = 120 , 0
MCU = atmega16a              program/data size = 112 , 0
MCU = atmega16c1             compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: crtm16c1.o
MCU = atmega16hva            program/data size = 112 , 0
MCU = atmega16hva2           program/data size = 116 , 0
MCU = atmega16hvb            program/data size = 144 , 0
MCU = atmega16m1             program/data size = 152 , 0
MCU = atmega16u2             program/data size = 180 , 0
MCU = atmega16u4             program/data size = 200 , 0
MCU = atmega2560             program/data size = 286 , 0
MCU = atmega2561             program/data size = 262 , 0
MCU = atmega32               program/data size = 112 , 0
MCU = atmega323              program/data size = 108 , 0
MCU = atmega324a             program/data size = 152 , 0
MCU = atmega324p             program/data size = 152 , 0
MCU = atmega324pa            program/data size = 152 , 0
MCU = atmega325              program/data size = 120 , 0
MCU = atmega3250             program/data size = 128 , 0
MCU = atmega3250p            program/data size = 128 , 0
MCU = atmega325p             program/data size = 120 , 0
MCU = atmega328              program/data size = 132 , 0
MCU = atmega328p             program/data size = 132 , 0
MCU = atmega329              program/data size = 120 , 0
MCU = atmega3290             program/data size = 128 , 0
MCU = atmega3290p            program/data size = 128 , 0
MCU = atmega329p             program/data size = 120 , 0
MCU = atmega329pa            program/data size = 120 , 0
MCU = atmega32c1             program/data size = 152 , 0
MCU = atmega32hvb            program/data size = 144 , 0
MCU = atmega32m1             program/data size = 152 , 0
MCU = atmega32u2             program/data size = 180 , 0
MCU = atmega32u4             program/data size = 200 , 0
MCU = atmega32u6             program/data size = 180 , 0
MCU = atmega406              program/data size = 120 , 0
MCU = atmega48               program/data size = 80 , 0
MCU = atmega48a              program/data size = 80 , 0
MCU = atmega48p              program/data size = 80 , 0
MCU = atmega4hvd             compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: crtm4hvd.o
MCU = atmega64               program/data size = 168 , 0
MCU = atmega640              program/data size = 256 , 0
MCU = atmega644              program/data size = 140 , 0
MCU = atmega644a             program/data size = 152 , 0
MCU = atmega644p             program/data size = 152 , 0
MCU = atmega644pa            program/data size = 152 , 0
MCU = atmega645              program/data size = 120 , 0
MCU = atmega6450             program/data size = 128 , 0
MCU = atmega6450a            program/data size = 128 , 0
MCU = atmega6450p            program/data size = 128 , 0
MCU = atmega645a             program/data size = 120 , 0
MCU = atmega645p             program/data size = 120 , 0
MCU = atmega649              program/data size = 120 , 0
MCU = atmega6490             program/data size = 128 , 0
```

```
MCU = atmega6490a             program/data size = 128 , 0
MCU = atmega6490p             program/data size = 128 , 0
MCU = atmega649a              program/data size = 120 , 0
MCU = atmega649p              program/data size = 120 , 0
MCU = atmega64c1              program/data size = 152 , 0
MCU = atmega64hve             program/data size = 128 , 0
MCU = atmega64m1              program/data size = 152 , 0
MCU = atmega8                 program/data size = 66 , 0
MCU = atmega8515              program/data size = 62 , 0
MCU = atmega8535              program/data size = 70 , 0
MCU = atmega88                program/data size = 80 , 0
MCU = atmega88a               program/data size = 80 , 0
MCU = atmega88p               program/data size = 80 , 0
MCU = atmega88pa              program/data size = 80 , 0
MCU = atmega8hva              program/data size = 70 , 0
MCU = atmega8hvd              compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: crtm8hvd.o
MCU = atmega8u2               program/data size = 180 , 0
MCU = attiny11                compile error: /tmp/pyavrutils_ouzPGd/pyavrutils_1H9a3d.c:1: error: MC
MCU = attiny12                compile error: /tmp/pyavrutils_r5kvED/pyavrutils_lhUfpM.c:1: error: MC
MCU = attiny13                program/data size = 44 , 0
MCU = attiny13a               program/data size = 44 , 0
MCU = attiny15                compile error: /tmp/pyavrutils_cbIX9O/pyavrutils_KDe1_B.c:1: error: MC
MCU = attiny167               program/data size = 108 , 0
MCU = attiny22                program/data size = 30 , 0
MCU = attiny2313              program/data size = 62 , 0
MCU = attiny2313a             program/data size = 66 , 0
MCU = attiny24                program/data size = 58 , 0
MCU = attiny24a               program/data size = 58 , 0
MCU = attiny25                program/data size = 54 , 0
MCU = attiny26                program/data size = 48 , 0
MCU = attiny261               program/data size = 62 , 0
MCU = attiny261a              program/data size = 62 , 0
MCU = attiny28                compile error: /tmp/pyavrutils_bXSUZR/pyavrutils_ykZmmN.c:1: error: MC
MCU = attiny4313              program/data size = 70 , 0
MCU = attiny43u               program/data size = 60 , 0
MCU = attiny44                program/data size = 62 , 0
MCU = attiny44a               program/data size = 62 , 0
MCU = attiny45                program/data size = 58 , 0
MCU = attiny461               program/data size = 66 , 0
MCU = attiny461a              compile error: Known MCU names:
MCU = attiny48                program/data size = 68 , 0
MCU = attiny84                program/data size = 62 , 0
MCU = attiny85                program/data size = 58 , 0
MCU = attiny861               program/data size = 66 , 0
MCU = attiny861a              program/data size = 66 , 0
MCU = attiny87                program/data size = 68 , 0
MCU = attiny88                program/data size = 68 , 0
MCU = atxmega128a1            program/data size = 568 , 0
MCU = atxmega128a3            program/data size = 546 , 0
MCU = atxmega128d3            program/data size = 514 , 0
MCU = atxmega16a4             program/data size = 404 , 0
MCU = atxmega16d4             program/data size = 392 , 0
MCU = atxmega192a3            program/data size = 546 , 0
MCU = atxmega192d3            program/data size = 514 , 0
MCU = atxmega256a3            program/data size = 546 , 0
MCU = atxmega256a3b           program/data size = 546 , 0
MCU = atxmega256d3            program/data size = 514 , 0
MCU = atxmega32a4             program/data size = 412 , 0
```

```
MCU = atxmega32d4              program/data size = 392 , 0
MCU = atxmega64a1              program/data size = 564 , 0
MCU = atxmega64a3              program/data size = 542 , 0
MCU = atxmega64d3              program/data size = 510 , 0
MCU = avr1                     compile error: /tmp/pyavrutils_828PWC/pyavrutils_uVa5CI.c:1: error: MC
MCU = avr2                     program/data size = 0 , 0
MCU = avr25                    program/data size = 0 , 0
MCU = avr3                     compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: avr:31 arch
MCU = avr31                    compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: avr:31 arch
MCU = avr35                    compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: avr:35 arch
MCU = avr4                     program/data size = 0 , 0
MCU = avr5                     program/data size = 0 , 0
MCU = avr51                    compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: avr:51 arch
MCU = avr6                     compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = avrxmega2                compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = avrxmega3                compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = avrxmega4                compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = avrxmega5                compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = avrxmega6                compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = avrxmega7                compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: skipping in
MCU = m3000f                   compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: crtm3000f.o
MCU = m3000s                   compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: crtm3000s.o
MCU = m3001b                   compile error: /usr/lib/gcc/avr/4.3.5/../../../avr/bin/ld: crtm3001b.o
```

CHAPTER

FIVE

BUILD TESTS

Code:

```
void setup()
{
}

void loop()
{
}
```

Results:

index	board	min
1	atmega8	OK
2	bt	OK
3	bt328	OK
4	diecimila	OK
5	fio	OK
6	lilypad	OK
7	lilypad328	OK
8	mega	OK
9	mega2560	OK
10	metaboard	OK
11	mini	OK
12	pro	OK
13	pro328	OK
14	pro5v	OK
15	pro5v328	OK
16	uno	OK
17	arduino_OrangutanSVP1284	OK
18	arduino_amber128	OK
19	arduino_android2561	OK
20	arduino_android2561_16	OK
21	arduino_at90can128	OK
22	arduino_at90can32	OK
23	arduino_at90can64	OK
24	arduino_at90usb162	OK
25	arduino_at90usb646	OK
26	arduino_at90usb647	OK
27	arduino_at90usbkey	OK

Continued on next page

Table 5.1 – continued from previous page

28	arduino_atmega16	OK
29	arduino_atmega165	OK
30	arduino_atmega3290p	OK
31	arduino_atmega8515	OK
32	arduino_atmega8535	OK
33	arduino_attiny2313	OK
34	arduino_attiny26	OK
35	arduino_attiny45	OK
36	arduino_attiny85	OK
37	arduino_bahbots1284p	OK
38	arduino_butterfly	OK
39	arduino_cerebot_plus	OK
40	arduino_cerebotii	OK
41	arduino_digilent_explorer	OK
42	arduino_duino644	OK
43	arduino_duino644p	OK
44	arduino_gator	OK
45	arduino_illuminato	OK
46	arduino_penguino_avr	OK
47	arduino_teensy2_ser	OK
48	arduino_teensypp2_ser	OK
49	arduino_wiring1281	OK
50	atmega168	OK
51	atmega328	OK
52	atmega48	OK
53	atmega640	OK
54	atmega8	OK
55	atmega88	OK
56	bt	OK
57	bt328	OK
58	diecimila	OK
59	dvk90can1	OK
60	ecavr_atmega32	OK
61	fio	OK
62	lilypad	OK
63	lilypad328	OK
64	mega	OK
65	mega1280stk500v2	OK
66	mega2560stk500v2	OK
67	mini	OK
68	pro	OK
69	pro328	OK
70	pro5v	OK
71	pro5v328	OK
72	stk502	OK
73	stk525	OK
74	stk525_647	OK

Board configuration:

index	package	id	name	MCU
				Continu

Table 5.2 – continued from previous page

1	arduino	atmega8	Arduino NG or older w/ ATmega8	atmega8
2	arduino	bt	Arduino BT w/ ATmega168	atmega168
3	arduino	bt328	Arduino BT w/ ATmega328	atmega328p
4	arduino	diecimila	Arduino Diecimila, Duemilanove, or Nano w/ ATmega168	atmega168
5	arduino	fio	Arduino Fio	atmega328p
6	arduino	lilypad	LilyPad Arduino w/ ATmega168	atmega168
7	arduino	lilypad328	LilyPad Arduino w/ ATmega328	atmega328p
8	arduino	mega	Arduino Mega (ATmega1280)	atmega1280
9	arduino	mega2560	Arduino Mega 2560	atmega2560
10	arduino	metaboard	Metaboard	atmega168
11	arduino	mini	Arduino Mini	atmega168
12	arduino	pro	Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168	atmega168
13	arduino	pro328	Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328	atmega328p
14	arduino	pro5v	Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168	atmega168
15	arduino	pro5v328	Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328	atmega328p
16	arduino	uno	Arduino Uno	atmega328p
17	arduino-extras	arduino_OrangutanSVP1284	Arduino-Orangutan SVP-1284	atmega1284
18	arduino-extras	arduino_amber128	Arduino-Amber 128 14.7456 Mhz	atmega128
19	arduino-extras	arduino_android2561	Arduino-Android 2561 8Mhz	atmega2561
20	arduino-extras	arduino_android2561_16	Arduino-Android 2561 16Mhz	atmega2561
21	arduino-extras	arduino_at90can128	AT90CAN128 development board NHL (arduino core)	at90can128
22	arduino-extras	arduino_at90can32	at90can32 (arduino core)	at90can32
23	arduino-extras	arduino_at90can64	at90can64 (arduino core)	at90can64
24	arduino-extras	arduino_at90usb162	Arduino-at90usb162	at90usb162
25	arduino-extras	arduino_at90usb646	Arduino-at90usb646	at90usb646
26	arduino-extras	arduino_at90usb647	Arduino-at90usb647	at90usb647
27	arduino-extras	arduino_at90usbkey	Arduino-at90usbkey	at90usb1287
28	arduino-extras	arduino_atmega16	Arduino-Atmega16	atmega16
29	arduino-extras	arduino_atmega165	Arduino-Atmega165	atmega165
30	arduino-extras	arduino_atmega3290p	Arduino-Atmega3290p	atmega3290
31	arduino-extras	arduino_atmega8515	Arduino-ATmega8515	atmega8515
32	arduino-extras	arduino_atmega8535	Arduino-Test-Atmega8535	atmega8535
33	arduino-extras	arduino_attiny2313	Arduino-ATtiny2313	attiny2313
34	arduino-extras	arduino_attiny26	Arduino-ATtiny26	attiny26
35	arduino-extras	arduino_attiny45	Arduino-ATtiny45	attiny45
36	arduino-extras	arduino_attiny85	Arduino-ATtiny85	attiny85
37	arduino-extras	arduino_bahbots1284p	Arduino-BahBots 1284p	atmega1284
38	arduino-extras	arduino_butterfly	Arduino-Butterfly stk500	atmega169
39	arduino-extras	arduino_cerebot_plus	Arduino-Cerebot Plus	atmega2560
40	arduino-extras	arduino_cerebotii	Arduino-Cerebot II atemga64	atmega64
41	arduino-extras	arduino_digilent_explorer	Arduino-Digilent I/O Explorer USB	atmega165p
42	arduino-extras	arduino_duino644	Arduino-Duino 644	atmega644
43	arduino-extras	arduino_duino644p	Arduino-Duino 644P	atmega644p
44	arduino-extras	arduino_gator	Arduino-Rugged Circuits Gator Board	atmega324p
45	arduino-extras	arduino_illuminato	Arduino-illuminato	atmega645
46	arduino-extras	arduino_penguino_avr	Arduino-Penguino AVR	atmega32
47	arduino-extras	arduino_teensy2_ser	Arduino-Teensy 2.0 (USB Serial)	atmega32u4
48	arduino-extras	arduino_teensypp2_ser	Arduino-Teensy++ 2.0 (USB Serial)	at90usb1286
49	arduino-extras	arduino_wiring1281	Arduino-Wiring 1281	atmega1281
50	arduino-extras	atmega168	Arduino NG or older w/ ATmega168	atmega168
51	arduino-extras	atmega328	Arduino Duemilanove or Nano w/ ATmega328	atmega328p
52	arduino-extras	atmega48	Arduino Atmega48	atmega48

Continu

Table 5.2 – continued from previous page

53	arduino-extras	atmega640	Arduino atmega640	atmega640
54	arduino-extras	atmega8	Arduino NG or older w/ ATmega8	atmega8
55	arduino-extras	atmega88	Atmega88	atmega88p
56	arduino-extras	bt	Arduino BT w/ ATmega168	atmega168
57	arduino-extras	bt328	Arduino BT w/ ATmega328	atmega328p
58	arduino-extras	diecimila	Arduino Diecimila, Duemilanove, or Nano w/ ATmega168	atmega168
59	arduino-extras	dvk90can1	STK500 w/DVK90CAN1 - AT90can128 (Arduino Core)	at90can128
60	arduino-extras	ecavr_atmega32	Embedded market atmega32	atmega32
61	arduino-extras	fio	Arduino Fio	atmega328p
62	arduino-extras	lilypad	LilyPad Arduino w/ ATmega168	atmega168
63	arduino-extras	lilypad328	LilyPad Arduino w/ ATmega328	atmega328p
64	arduino-extras	mega	Arduino Mega	atmega1280
65	arduino-extras	mega1280stk500v2	Arduino Mega1280 stk500v2	atmega1280
66	arduino-extras	mega2560stk500v2	Arduino Mega2560 stk500v2	atmega2560
67	arduino-extras	mini	Arduino Mini	atmega168
68	arduino-extras	pro	Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168	atmega168
69	arduino-extras	pro328	Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328	atmega328p
70	arduino-extras	pro5v	Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168	atmega168
71	arduino-extras	pro5v328	Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328	atmega328p
72	arduino-extras	stk502	STK500 w/STKk502 - ATmega169 (Arduino Core)	atmega169
73	arduino-extras	stk525	STK500 w/STK525 - at90usb1287 (Arduino Core)	at90usb1287
74	arduino-extras	stk525_647	STK500 w/STK525 - at90usb647 (Arduino Core)	at90usb647

CHAPTER

SIX

API

class pyavrutils.**AvrGcc**(*mcu='atmega168'*)

build(*sources=None*, *headers=None*)
sources can be file name or code: sources=['x.c','int main(){}'] or sources='int main(){}'

command_list(*sources*, *_opt=False*)
command line as list

error_text

ok

optimize_for_size()
http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=90752

http://www.avrfreaks.net/index.php?name=PNphpBB2&file=viewtopic&t=69813

optimize_no()
all options set to default

options_generated()

size()

targets

version()
avr-gcc version

class pyavrutils.**AvrSize**
wrapper for avr-size

ok

parse_output(*s*)
Example output:

Device: atmega2561

Program: 4168 bytes (1.6% Full) (.text + .data + .bootloader)

Data: 72 bytes (0.9% Full) (.data + .bss + .noinit)

run(*objfile*, *mcu*)

class pyavrutils.**Arduino**(*board='pro'*, *hwpack='arduino'*, *mcu=None*, *f_cpu=None*, *extra_lib=None*, *ver=None*, *home='auto'*)
wrapper for arscons

build(*sources=None*)

command_list()
command line as list

error_text

mcu_compiler()

ok

size()

CHAPTER

SEVEN

DEVELOPMENT

7.1 Tools

1. setuptools
2. Paver
3. nose
4. ghp-import
5. pyflakes
6. pychecker
7. paved fork
8. Sphinx
9. sphinxcontrib-programscreenshot
10. sphinxcontrib-paverutils
11. `autorun` from sphinx-contrib (there is no simple method, you have to download/unpack/setup)

7.2 Install on ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install python-paver
sudo apt-get install python-nose
sudo easy_install ghp-import
sudo apt-get install pyflakes
sudo apt-get install pychecker
sudo easy_install https://github.com/ponty/paved/zipball/master
sudo apt-get install scrot
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install python-imaging
sudo apt-get install python-sphinx
sudo easy_install sphinxcontrib-programscreenshot
sudo easy_install sphinxcontrib-programoutput
sudo easy_install sphinxcontrib-paverutils
```

7.3 Tasks

Paver is used for task management, settings are saved in pavement.py. Sphinx is used to generate documentation.

print paver settings:

```
paver printoptions
```

clean generated files:

```
paver clean
```

generate documentation under *docs/_build/html*:

```
paver cog pdf html
```

upload documentation to github:

```
paver ghpages
```

run unit tests:

```
paver nose
#or
nosetests --verbose
```

check python code:

```
paver pyflakes
paver pychecker
```

generate python distribution:

```
paver sdist
```

upload python distribution to PyPI:

```
paver upload
```

CHAPTER

EIGHT

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

INDEX