Bare Metal Assembly Programming

- Input Tests

Basedn on “Baking Pi: Operating Systems Development” by Alex Chadwick

Young Won Lim
9/19/15

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice.

LED Test

```
.section .init
.globl _start
_start:

ldr r0,=0x20200000

mov r1,#1
lsl r1,#18

str r1,[r0,#4]

mov r1,#1
lsl r1,#16

str r1,[r0,#40]

loop$:
b loop$
```

https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html

GAS Directives

.global *symbol*

.global makes the symbol visible to **ld**.

If you define *symbol* in your partial program,its value is made available to other partial programs that are linked with it.
Otherwise, *symbol* takes its attributes from a symbol of the same name from another file

.section *name*

Use the **.section** directive to assemble the following code into a section named *name*.

https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html

Sections

named sections / text section / data section
hold your program.
as and **ld** treat them as separate but equal sections.

the text section

- is to be unalterable.
- is often shared among processes
- contains instructions, constants and the like

the data section

- is usually alterable
- C variables would be stored in the data section.

bss section

- contains zeroed bytes
- hold uninitialized variables or common storage.
- The length of each partial program's bss section
- no need to store explicit zeros in the object file
- to eliminate those explicit zeros from object files.

https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html

Sections

absolute section

- Address 0 of this section
  is always “relocated” to runtime address 0.
- to refer to an address that ld must not change
- “unrelocatable”:

undefined section

a catch-all for address references to objects not in the preceding sections.

https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html

LED Test

```
.section .init
.globl _start
_start:

ldr r0,=0x20200000

mov r1,#1
lsl r1,#18

str r1,[r0,#4]

mov r1,#1
lsl r1,#16
```

```
loop$:

str r1,[r0,#40]

mov r2,#0x3F0000
wait1$:
    sub r2,#1
    cmp r2,#0
    bne wait1$

str r1,[r0,#28]

mov r2,#0x3F0000
wait2$:
    sub r2,#1
    cmp r2,#0
    bne wait2$

b loop$
```

https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html

LED Test

```
.section .init
.globl _start
_start:

b main
.section .text

main:

mov sp,#0x8000

pinNum .req r0
pinFunc .req r1
mov pinNum,#16
mov pinFunc,#1
bl SetGpioFunction
.unreq pinNum
.unreq pinFunc

loop$:
pinNum .req r0
pinVal .req r1
mov pinNum,#16
mov pinVal,#0
bl SetGpio
.unreq pinNum
.unreq pinVal
```

```
decr .req r0
mov decr,#0x3F0000
wait1$:
    sub decr,#1
    teq decr,#0
    bne wait1$
.unreq decr

pinNum .req r0
pinVal .req r1
mov pinNum,#16
mov pinVal,#1
bl SetGpio
.unreq pinNum
.unreq pinVal

decr .req r0
mov decr,#0x3F0000
wait2$:
    sub decr,#1
    teq decr,#0
    bne wait2$
.unreq decr

b loop$
```

https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html

Address Type Casting (1)aaaaaaaa

References

[1] http://wiki.osdev.org/ARM_RaspberryPi_Tutorial_C
[2] http://blog.bobuhiro11.net/2014/01-13-baremetal.html
[3] http://www.valvers.com/open-software/raspberry-pi/
[4] https://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/os/downloads.html