

Trabajo Práctico 1

Agustin Alejandro Linari, *Padrón Nro. 81.783*
agustinlinari@gmail.com

Juan Ignacio López Pecora, *Padrón Nro. 84.700*
jlopezpecora@gmail.com

Pablo Daniel Sívori, *Padrón: 84.026*
sivoridaniel@gmail.com

2° Cuatrimestre de 2016
66.20 Organizacion de Computadoras
Facultad de Ingenieria, Universidad de Buenos Aires

22 de octubre de 2016

Resumen

En el presente trabajo utilizamos el conjunto de instrucciones MIPS y el concepto de ABI para resolver parte de la lógica del programa realizado en el trabajo práctico 0.

Índice

I	Desarrollo	3
1.	Introduccion	3
2.	Build	3
3.	Diseño e Implementación del Programa	3
3.1.	Stack Frame	3
4.	Corridas de Programa	5
4.1.	Ejemplos	5
4.2.	Pruebas	5
II	Apendice	6
A.	Codigo fuente	6
B.	Enunciado original	7

Parte I

Desarrollo

1. Introduccion

El objetivo del presente trabajo práctico es familiarizarse con el código de instrucciones MIPS 32. Para ello implementaremos la lógica de cómputo del fractal con dicho código de instrucciones. Finalmente compilaremos el programa en el emulador GXemul para poder obtener el código de instrucciones Mips32.

2. Build

El correspondiente informe se puede construir utilizando el make con la etiqueta doc la cual borra y genera el informe en formato pdf.

3. Diseño e Implementación del Programa

El código fuente del programa se puede encontrar en el anexo A.

3.1. Stack Frame

A continuación mostramos los diagrama de stack frame de las funciones implementadas en MIPS32.

int buff_write(int fd, char* buf, int size)	
ABA (caller)	64 size
	60 str
	56 fd
SRA	52 ra
	48 gp
	44 fp
	40 s6
	36 s5
	32 s4
	28 s3
	24 s2
	20 s1
ABA	16 s0
	12
	8
	4
	0 a0 (fd)

Figura 1: Stack1

int buff_flush(int fd)		
ABA (caller)		
	16	fd
SRA	12	gp
	8	fp
LTA	4	
	0	result

Figura 2: Stack2

int print_int(int n, int fd)		
ABA (caller)	44	fd
	40	n
SRA	36	
	32	ra
	28	fp
	24	sp
LTA	20	
	16	r
ABA	12	a3
	8	a2
	4	a1
	0	a0

Figura 3: Stack3

int mips32_plot(param_t* params)		
ABA (caller)	88	params (param_t*)
SRA	84	
	80	ra
	76	gp
	72	fp
LTA	68	
	64	cpi
	60	cpr
	56	res
	52	c
	48	y
	44	x
	40	absz
	36	si
	32	sr
	28	zi
	24	zr
	20	ci
	16	cr
ABA	12	a3
	8	a2
	4	a1
	0	a0

Figura 4: Stack4

4. Corridas de Programa

4.1. Ejemplos

4.2. Pruebas

Parte II

Apendice

A. Código fuente

B. Enunciado original

Universidad de Buenos Aires - FIUBA
66.20 Organización de Computadoras
Trabajo práctico 1: conjunto de instrucciones MIPS
2º cuatrimestre de 2016

\$Date: 2016/10/02 21:23:43 \$

1. Objetivos

Familiarizarse con el conjunto de instrucciones MIPS y el concepto de ABI, extendiendo un programa que resuelva el problema descrito en la sección 4.

2. Alcance

Este trabajo práctico es de elaboración grupal, evaluación individual, y de carácter obligatorio para todos alumnos del curso.

3. Requisitos

El trabajo deberá ser entregado personalmente, en la fecha estipulada, con una carátula que contenga los datos completos de todos los integrantes, un informe impreso de acuerdo con lo que mencionaremos en la sección 5, y con una copia digital de los archivos fuente necesarios para compilar el trabajo.

4. Descripción

Se trata de un modificar un programa que dibuje el conjunto de Julia y sus vecindades introducido en el *TP0* [1], en el cual la lógica de cómputo del fractal deberá tener soporte nativo para MIPS32 sobre NetBSD/pmax.

El código fuente con la versión inicial del programa, se encuentra disponible en [2]. El mismo deberá ser considerado como punto de partida de todas las implementaciones.

4.1. Soporte para MIPS

El entregable producido en este trabajo deberá implementar la lógica de cómputo del fractal en assembly MIPS32, con soporte nativo para NetBSD/pmax.

Para ello, cada grupo deberá tomar el código fuente de base para este TP, [2], y reescribir la función `mips32_plot()` sin cambiar su API. Esta función está ubicada en el archivo `mips32_plot.c`.

4.2. Casos de prueba

El informe trabajo práctico deberá incluir una sección dedicada a verificar el funcionamiento del código implementado. Para ello, será necesario escribir pruebas orientadas a probar el programa completo, ejercitando los casos más comunes de funcionamiento, los casos de borde, y también casos de error.

4.3. Compilación

El código fuente provisto por la cátedra provee los makefiles necesarios para compilar el ejecutable a partir de la versión en C con el archivo `mips32_plot.c`. Para poder compilar el código desarrollado deberán cambiar la definición en el archivo `Makefile.in` la línea número 6:

```
SRCS = mips32_plot.c main.c mygetopt_long.c
```

por

```
SRCS = mips32_plot.S main.c mygetopt_long.c
```

Luego deberán invocar la siguiente secuencia de comandos para limpiar los archivos temporales y generar los nuevos Makefiles:

```
$ make clean
$ make makefiles
$ make
```

4.4. Detalles de la implementación

Para optimizar los accesos a las llamadas a servicio del sistema (`syscalls`), deben utilizar un buffer de n bytes para escribir los datos de salida para luego ser enviados al archivo de salida. El tamaño n debe ser parametrizable mediante un `#define`.

5. Informe

El informe, a entregar en formarto impreso y digital¹ deberá incluir:

- Documentación relevante al diseño e implementación del código desarrollado para adaptar el programa. Incluir el diagrama de *stack frame* de las funciones implementadas en MIPS32.
- Documentación relevante al proceso de compilación: cómo obtener el ejecutable a partir de los archivos fuente. Especificar modificaciones realizadas a los archivos provistos por la cátedra si es que los hubo.
- Las corridas de prueba, con los comentarios pertinentes.²
- El código fuente, en lenguaje C (y MIPS32 donde corresponda)
- Este enunciado.

¹En CD, DVD o memoria flash.

²Las pruebas provistas deben ejecutarse correctamente en NetBSD sobre MIPS32 sin modificación alguna.

6. Fecha de entrega

La fecha de vencimiento será el Martes 01/11.

Referencias

- [1] Trabajo Práctico 0, 2do cuatrimestre de 2016.
<https://groups.yahoo.com/neo/groups/orga-comp/files/TPs/tp0-2016-2q.pdf>.
- [2] Código fuente con el esqueleto del trabajo práctico.
https://drive.google.com/open?id=0B93s6e6NY_j1TFV2TFBqbUNKZ3M.