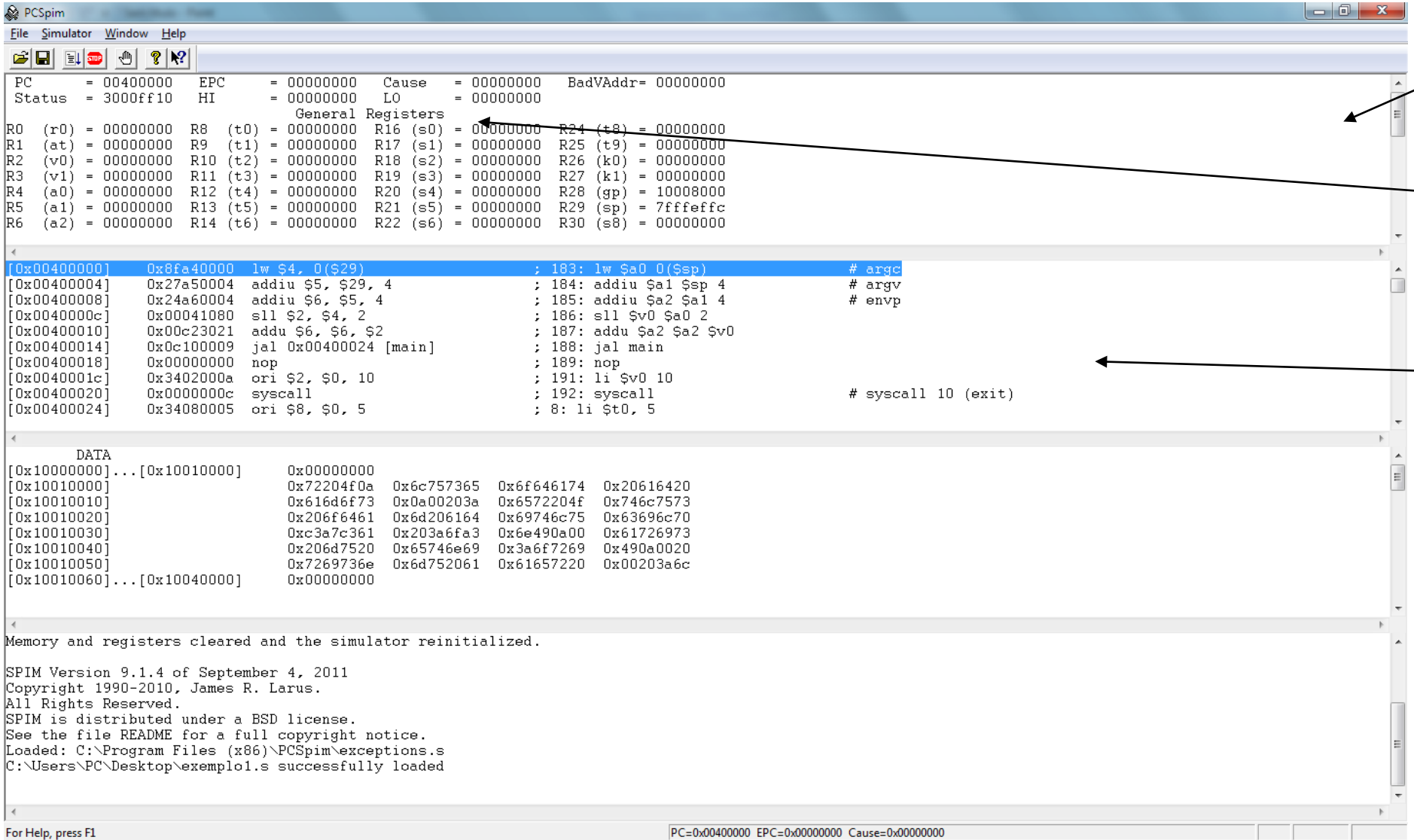
PC SPIM

Visão Preliminar

PC SPIM



Registradores

Registradores de uso geral (Inteiros e Caracteres)

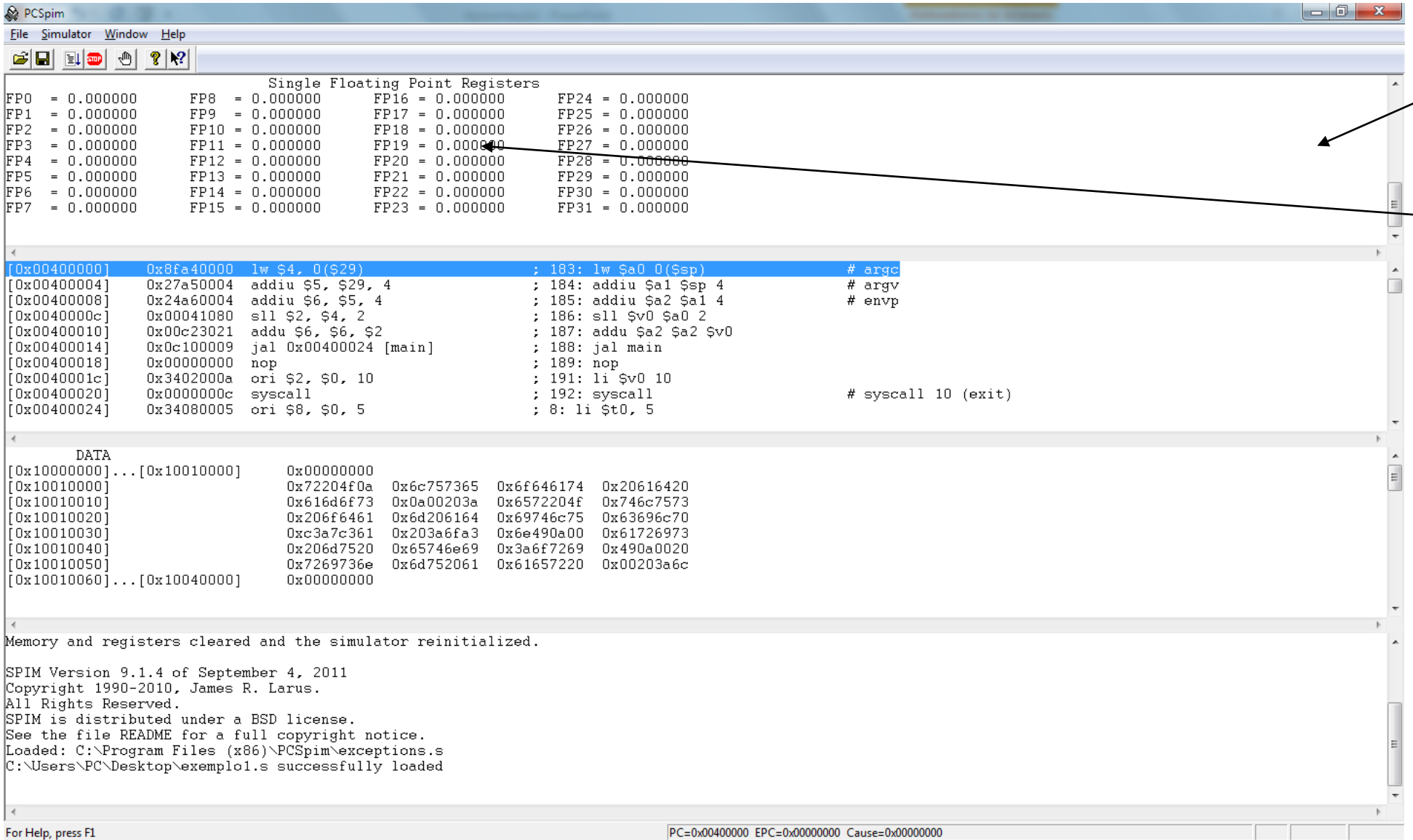
Set de Instruções

PC SPIM

Serão livres para utilização:
- Registradores de t0 a t9
- Registradores de s0 a s7

Serão reservados:
- v0, para códigos de leitura e escrita
- a0, para sequência de caracteres oriundas do .data

PC SPIM



Registradores

Registradores de ponto flutuante de precisão *Single* (reais)

PC SPIM

Serão livres para utilização:
- Registradores de f1 a f11
- Registradores de f13 a f31

Serão reservados:
- f12, para escrita de pontos flutuantes
- f0, para leitura de pontos flutuantes

PC SPIM

Primeiros comandos:

- Atribuição de inteiros:
  li pos, valor
  Ex.: li $t0, 3

- Atribuição de reais:
  li.s pos, valor
  Ex.: li $f1, 5.3

- Operações Fundamentais Inteiros:
  Soma: add
  Subtração: sub
  Multiplicação: mul
  Divisão: div
  Resto: rem

- Exemplo: Operação de $t4 e $t5 com resultado em $t0:
  add $t0, $t4, $t5
  sub $t0, $t4, $t5
  mul $t0, $t4, $t5
  div $t0, $t4, $t5
  rem $t0, $t4, $t5

- Operações Fundamentais Reais:
  Soma: add.s
  Subtração: sub.s
  Multiplicação: mul.s
  Divisão: div.s

- Exemplo: Operação de $f4 e $f5 com resultado em $f1:
  add.s $f1, $f4, $f5
  sub.s $f1, $f4, $f5
  mul.s $f1, $f4, $f5
  div.s $f1, $f4, $f5

PC SPIM

Códigos para $v0:

1: Escrita de Inteiros
2: Escrita de Reais (*Single*)
4: Escrita de Caracteres
5: Leitura de Inteiros
6: Leitura de Reais (*Single*)

- Escrita de caracteres do .data:

```
li $v0, 4
la $a0, texto1
syscall
```

- Escrita de Inteiros:

```
li $v0, 1
add $a0, $t2, $zero
syscall
```

*Passando para a0 o valor do registrador que será impresso

- Escrita de Reais:

```
li $v0, 2
mov.s $f12, $f3
syscall
```

*Passando para f0 o valor do registrador que será impresso

- Leitura de Inteiros:

```
li $v0, 5
syscall
add $t3, $v0, $zero
```

* Passando para $t3 o valor lido em a0

- Leitura de Reais:

```
li $v0, 6
syscall
mov.s $f31,$f0
```

* Passando para $f31 o valor lido em $f0

PC SPIM

Para criar estruturas:

- Jump:
  j label

- Branches:

Branch se igual	beq $s1, $s2, L	if($s1 == $s2) go to L
Branch se maior ou igual	bge $rs,$rt,Label	if($rs>=$rt) PC=Label
Branch se maior	bgt $rs,$rt,Label	if($rs>$rt) PC=Label
Branch se módulo maior	bgtu $rs,$rt,Label	if(|$rs\|>=\|$rt|) PC=Label
Branch se maior que zero	bgtz $rs,$rt,Label	if($rs>0) PC=Label
Branch se menor ou igual	ble $rs,$rt,Label	if($rs<=$rt) PC=Label
Branch se menor	blt $rs,$rt,Label	if($rs<$rt) PC=Label
Branch se diferente	bne $s1, $s2, L	if($s1 != $s2) go to L