

Assembly Language Program in a single file (p1.asm)

main:

```
sw $31, -4($30)
lis $29
.word fred
jalr $29
```

```
lis $29
.word derf
jalr $29
```

```
lw $31, -4($30)
jr $31
```

x: .word 0

fred:

```
; store $1 in x
lis $29
.word x
sw $1, 0($29)
```

```
; store $2 in y
lis $29
.word y
sw $2, 0($29)
jr $31
```

derf:

```
; load x into $2
lis $29
.word x
lw $2, 0($29)
```

```
; load y into $1
lis $29
.word y
lw $1, 0($29)
```

```
jr $31
y: .word 0
```

What problems occur if split in the natural way?

p1main.asm	p2fred.asm	p2derf.asm
<pre>main:sw \$31, -4(\$30) lis \$29 .word fred jalr \$29 lis \$29 .word derf jalr \$29 lw \$31, -4(\$30) jr \$31 x: .word 0</pre>	<pre>fred: ; store \$1 in x lis \$29 .word x sw \$1, 0(\$29) ; store \$2 in y lis \$29 .word y sw \$2, 0(\$29) jr \$31</pre>	<pre>derf: ; load x into \$2 lis \$29 .word x lw \$2, 0(\$29) ; load y into \$1 lis \$29 .word y lw \$1, 0(\$29) jr \$31 y: .word 0</pre>

Importing and Exporting Symbols

p1main.asm	p2fred.asm	p2derf.asm
<pre>.import fred .import derf .export x main:sw \$31, -4(\$30) lis \$29 .word fred jalr \$29 lis \$29 .word derf jalr \$29 lw \$31, -4(\$30) jr \$31 x: .word 0</pre>	<pre>.import x .import y .export fred fred: ; store \$1 in x lis \$29 .word x sw \$1, 0(\$29) ; store \$2 in y lis \$29 .word y sw \$2, 0(\$29) jr \$31</pre>	<pre>.import x .export derf .export y derf: ; load x into \$2 lis \$29 .word x lw \$2, 0(\$29) ; load y into \$1 lis \$29 .word y lw \$1, 0(\$29) jr \$31 y: .word 0</pre>

Merl Code

<code>.import fred</code>	<code>000: 1000 0002 ; merl cookie</code>
<code>.import derf</code>	<code>004: 0000 007c ; overall length</code>
<code>.export x</code>	<code>008: 0000 0034 ; header + code</code>
<code>main:sw \$31, -4(\$30)</code>	<code>00c: afd0 fffc ; start of code</code>
<code>lis \$29</code>	<code>010: 0000 e814</code>
<code>.word fred</code>	<code>014: 0000 0000</code>
<code>jalr \$29</code>	<code>018: 03a0 0009</code>
<code>lis \$29</code>	<code>01c: 0000 e814</code>
<code>.word derf</code>	<code>020: 0000 0000</code>
<code>jalr \$29</code>	<code>024: 03a0 0009</code>
<code>lw \$31, -4(\$30)</code>	<code>028: 8fdf fffc</code>
<code>jr \$31</code>	<code>02c: 03e0 0008</code>
<code>x: .word 0</code>	<code>030: 0000 0000 ; end of code</code>
	<code>034: 0000 0005 ; export x [ESD]</code>
	<code>038: 0000 0030 ; location of x</code>
	<code>03c: 0000 0001 ; name length</code>
	<code>040: 0000 0078 ; 'x' in hex</code>
	<code>044: 0000 0011 ; import fred [ESR]</code>
	<code>048: 0000 0014 ; location where used</code>
	<code>04c: 0000 0004 ; length of name</code>
	<code>050: 0000 0066 ; 'f' in hex</code>
	<code>054: 0000 0072 ; 'r' in hex</code>
	<code>058: 0000 0065 ; 'e' in hex</code>
	<code>05c: 0000 0064 ; 'd' in hex</code>
	<code>060: 0000 0011 ; import derf [ESR]</code>
	<code>064: 0000 0020 ; location where used</code>
	<code>068: 0000 0004 ; length of name</code>
	<code>06c: 0000 0064 ; 'd' in hex</code>
	<code>070: 0000 0065 ; 'e' in hex</code>
	<code>074: 0000 0072 ; 'r' in hex</code>
	<code>078: 0000 0066 ; 'f' in hex</code>