APPENDIX B
MULTIPLE-PRECISION SHIFTS

This appendix gives examples of how multiple precision shifts can be programmed. A multiple-precision shift is initially defined to be a shift of an *n*-word quantity, where *n* > 1. The quantity to be shifted is contained in *n* registers. The shift amount is specified either by an immediate value in the instruction or by bits 27 to 31 of a register.

The examples shown below distinguish between the cases *n* = 2 and *n* > 2. If *n* = 2, the shift amount may be in the range 0 to 63, which are the maximum ranges supported by the shift instructions used. However if *n* > 2, the shift amount must be in the range 0 to 31, for the examples to yield the desired result. The specific instance shown for *n* > 2 is *n* = 3: extending those instruction sequences to larger *n* is straightforward, as is reducing them to the case *n* = 2 when the more stringent restriction on shift amount is met. For shifts with immediate shift amounts only the case *n* = 3 is shown, because the more stringent restriction on shift amount is always met.

In the examples it is assumed that GPRs 2 and 3 (and 4) contain the quantity to be shifted, and that the result is to be placed into the same registers. In all cases, for both input and result, the lowest-numbered register contains the highest-order part of the data and highest-numbered register contains the lowest-order part. For non-immediate shifts, the shift amount is assumed to be in bits 27 to 31 (32-bit mode) of GPR6. For immediate shifts, the shift amount is assumed to be greater than zero. GPRs 0 to 31 are used as scratch registers. For *n* > 2, the number of instructions required is 2N-1 (immediate shifts) or 3N-1 (non-immediate shifts).

In the following examples, let *n* be the number of words to be shifted.

Shift Left Immediate, *n* = 3 (Shift Amount < 32)

```
rlwinm      r2,r2,SH,0,31-SH
rlwimi      r2,r3,SH,32-SH,31
rlwinm      r3,r3,SH,0,31-SH
rlwimi      r3,r4,SH,32-SH,31
rlwinm      r4,r4,SH,0,31-SH
```

Shift Left, *n* = 2 (Shift Amount < 64)

```
subfic      r31,r6,32
slw         r2,r2,r6
srw         r0,r3,r31
or          r2,r2,r0
addic       r31,r6,r6
slw         r0,r3,r31
or          r2,r2,r0
slw         r3,r3,r6
```

Shift Left, *n* = 3 (Shift Amount < 32)

```
subfic      r31,r6,32
slw         r2,r2,r6
srw         r0,r3,r31
or          r2,r2,r0
slw         r3,r3,6
srw         r0,r4,r31
or          r3,r3,r0
slw         r4,r4,r6
```

Shift Right Immediate, *n* = 3 (Shift Amount < 32)

```
rlwinm      r4,r4,32-SH,SH,31
rlwimi      r4,r3,32-SH,0,SH-1
rlwinm      r3,r3,32-SH,SH,31
rlwimi      r3,r2,32-SH,0,SH-1
rlwinm      r2,r2,32-SH,SH,31
```

Shift Right, *n* = 2 (Shift Amount < 64)

```
subfic      r31,r6,32
srw         r3,r3,r6
slw         r0,r2,r31
or          r3,r3,r0
addic       r31,r6,-32
srw         r0,r2,r31
or          r3,r3,r0
srw         r2,r2,r6
```

Shift Right, *n* = 3 (Shift Amount < 32)

```
subfic      r31,r6,32
srw         r4,r4,r6
slw         r0,r2,r31
or          r4,r4,r0
srw         r31,r3,r6
slw         r0,r2,r31
or          r3,r3,r0
srw         r2,r2,r6
```

Shift Right Algebraic Immediate, *n* = 3 (Shift Amount < 32)

```
rlwinm      r4,r4,32-SH,SH,31
rlwimi      r4,r3,32-SH,0,SH-1
rlwinm      r3,r3,32-SH,SH,31
rlwimi      r3,r2,32-SH,0,SH-1
srawi       r2,r2,SH
```

Shift Right Algebraic, *n* = 2 (Shift Amount < 64)

```
subfic      r31,r6,32
srw         r3,r3,r6
slw         r0,r2,r31
or          r3,r3,r0
addic.      r31,r6,-32
sraw        r0,r2,r31
ble         $+8
ori         r3,r0,0
sraw        r2,r2,r6
```

Shift Right Algebraic, *n* = 3 (Shift Amount < 32)

```
subfic      r31,r6,32
srw         r4,r4,r6
slw         r0,r3,r31
or          r4,r4,r0
srw         r3,r3,r6
slw         r0,r2,r31
or          r3,r3,r0
sraw        r2,r2,r6
```