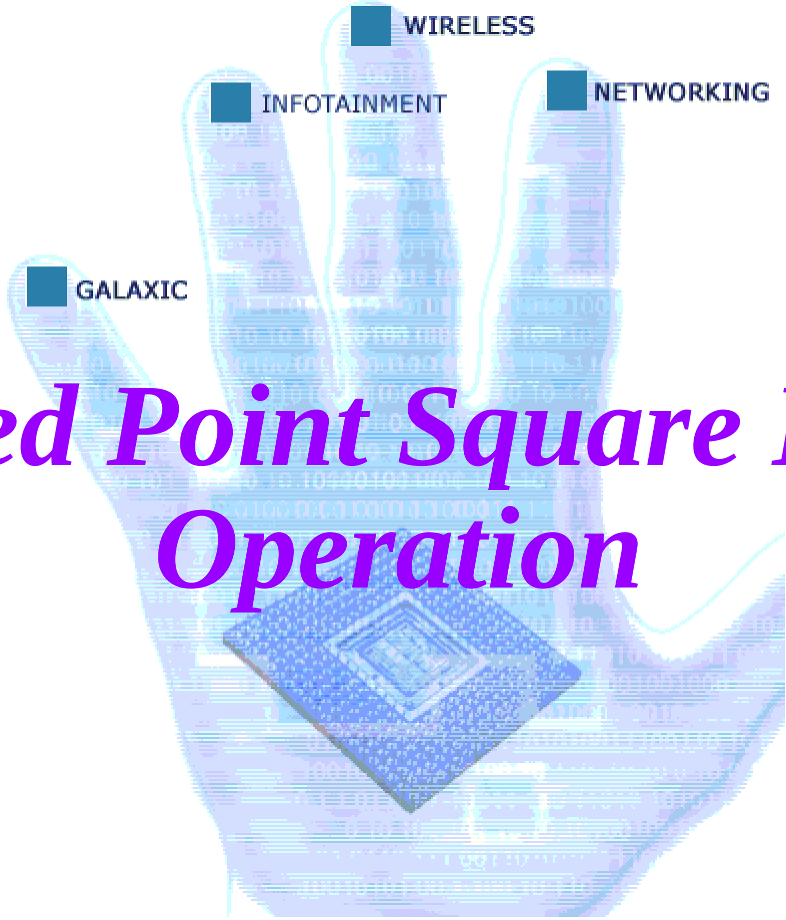
Fixed Point Square Root Operation



Square Root Operation

Abstract

Square root operation is considered a difficult operation to implement in hardware, will present a FPGA implementation of a 32 bit fixed-point square root based on the non-restoring algorithm square root.

The Non-Restoring Algorithm

This algorithm uses the two´s complement representation for the square root result, at each iteration the algorithm can genereta exact result value even in the last bit. there is not need to do the complex calculation as other methods. The exact remainder can be obtained immediately (with a little correction if it is negative). Assume that the radicand is an 32-bit unsigned number (denoted by D[31..0]).

The square root is denoted by Q[15..0]. R is the remainder

(R = D - (Q2)) which willbe denoted by R[16..0]

Square Root Operation

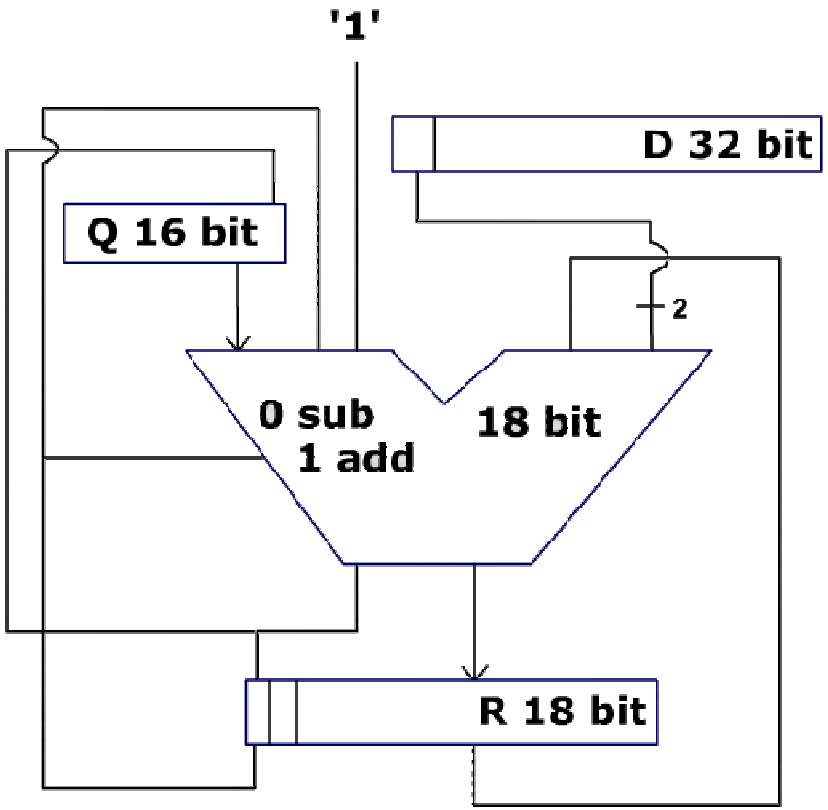
Hardware Design

The circuit was designed to use two shift registers (one is a shift 1-bit left, the other is shift 2-bit left), one normal register, and an adder. D is the radicand, Q is the solution and R is the remainder.

The size of each register (D,Q and R) and ALU can be determinated by de size of radicand register. If the radicand contain X bit, Q will be X/2, while ALU and R would be (X/2)+2, bit, and the iteration total number is (X/2)+1 cylce. In each cycle D will be shift left 2 bit and Q will be shift left 1 bit. The start up value of register Q and R is "0" and should be clear once the radicandis loaded into register D.

Square Root Operation

Datapath



Algorithm

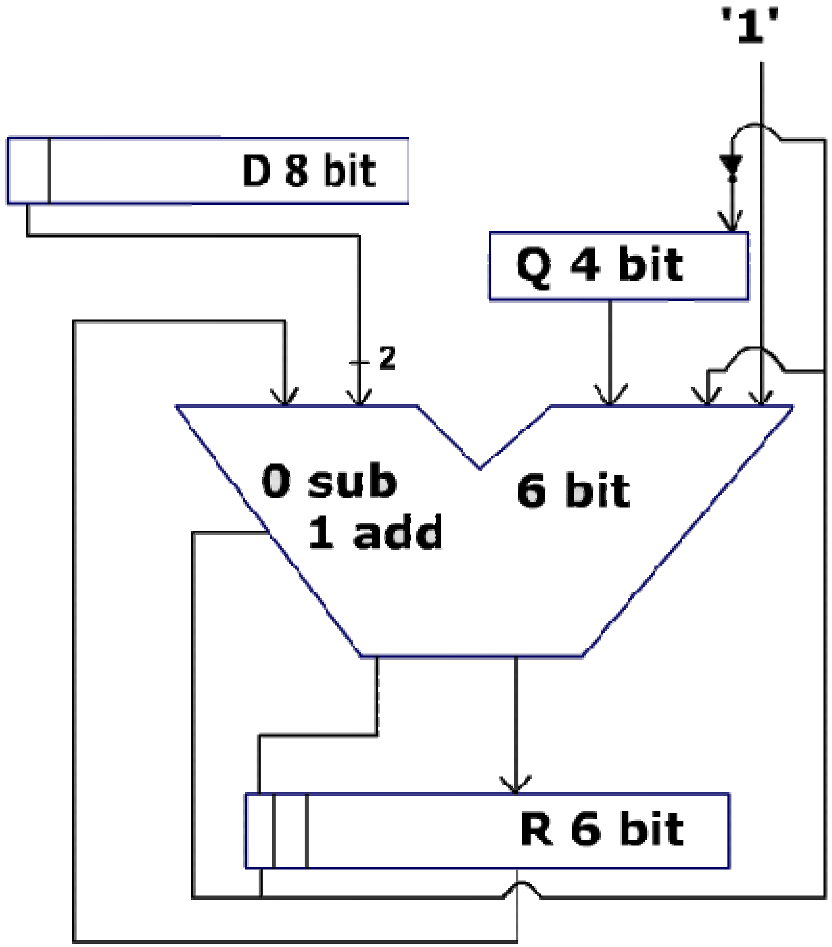
D be 32-bit unsigned integer
Q be 16-bit unsigned interger (Result)
R be 17-bit integer (R = D - Q2)

Algorithm
Q=0;R=0;

```
for i=15 to 0 do
 if (R >= 0)
   R= (R<<2) or (D >> (i+i) & 3);
   R= R - ((Q << 2) or 1);
 Else
  R= (R<<2) or (D >> (i+i) & 3);
  R= R - ((Q << 2) or 3);
 End if
 if (R >= 0)
 Q = (Q <<1) or 1;
 Else
 Q = (Q <<1) or 0;
End if
```

Square Root Operation

Really Datapath



Algorithm Rewrite

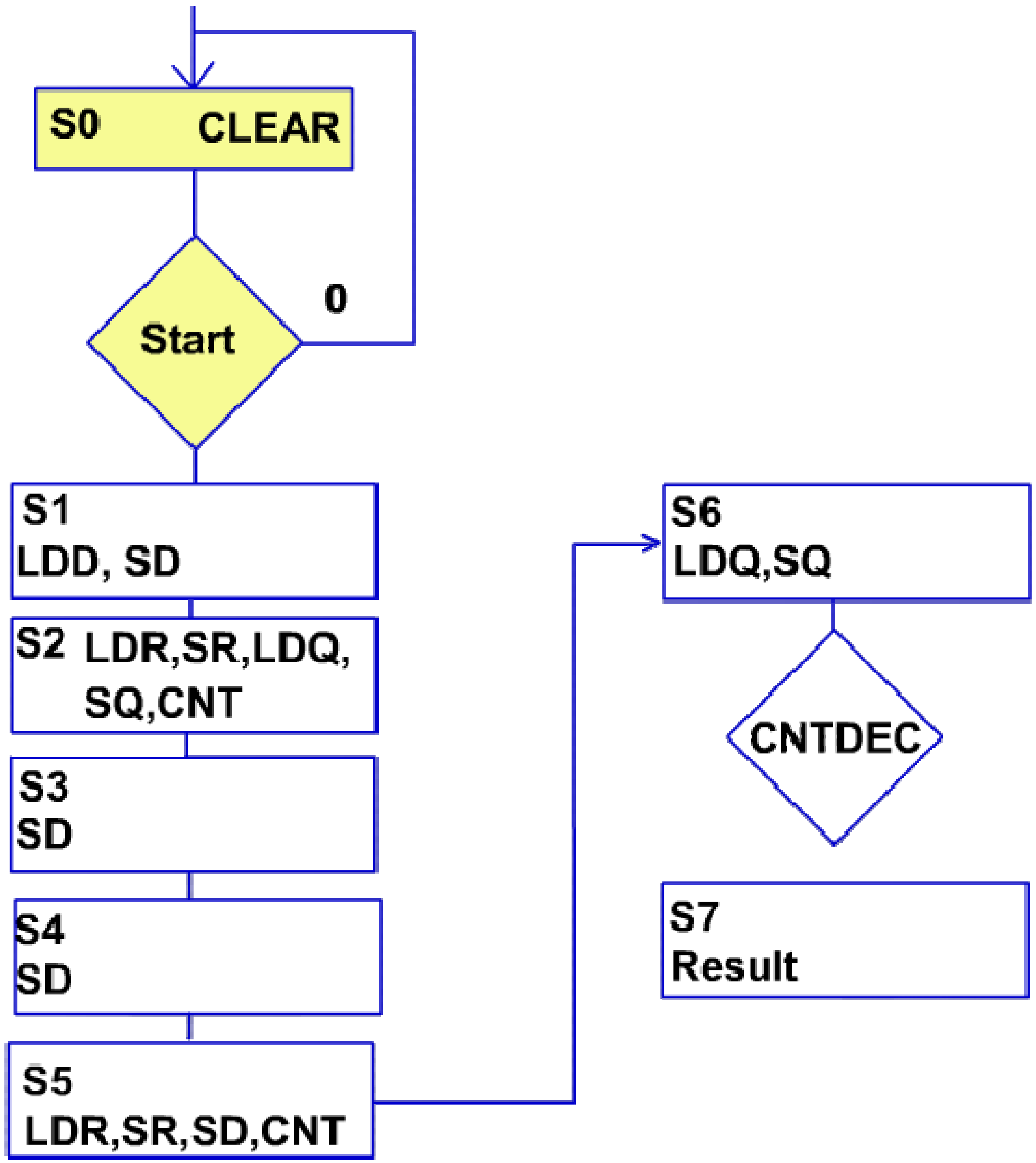
D be 8-bit unsigned integer
Q be 4-bit unsigned interger (Result)
R be 5-bit integer (R = D - Q2)

Algorithm
Q=0;R=0;

for i=3 to 0 do
 if (i=3)
 R= R or (D & 192);
 else
 R= (R<<2) or ((D << 2) & 192);
if (R >= 0)
   R= R - ((Q << 2) or 1); Q = (Q <<1)
   Q = Q or 1;
 Else
  R= R - ((Q << 2) or 3);Q = (Q <<1)
  Q = Q or 0;
 End if

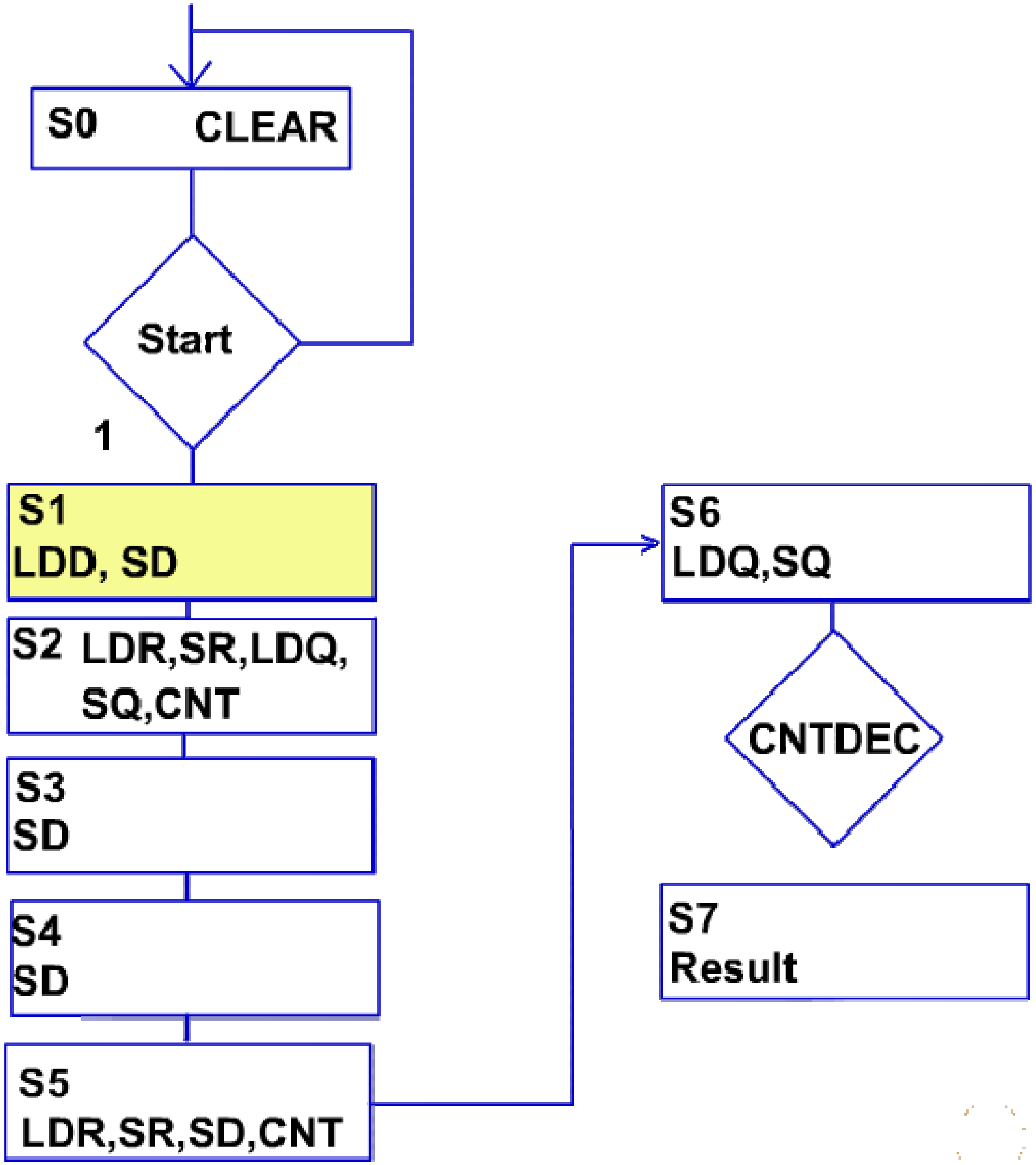
Square Root Operation

ASM



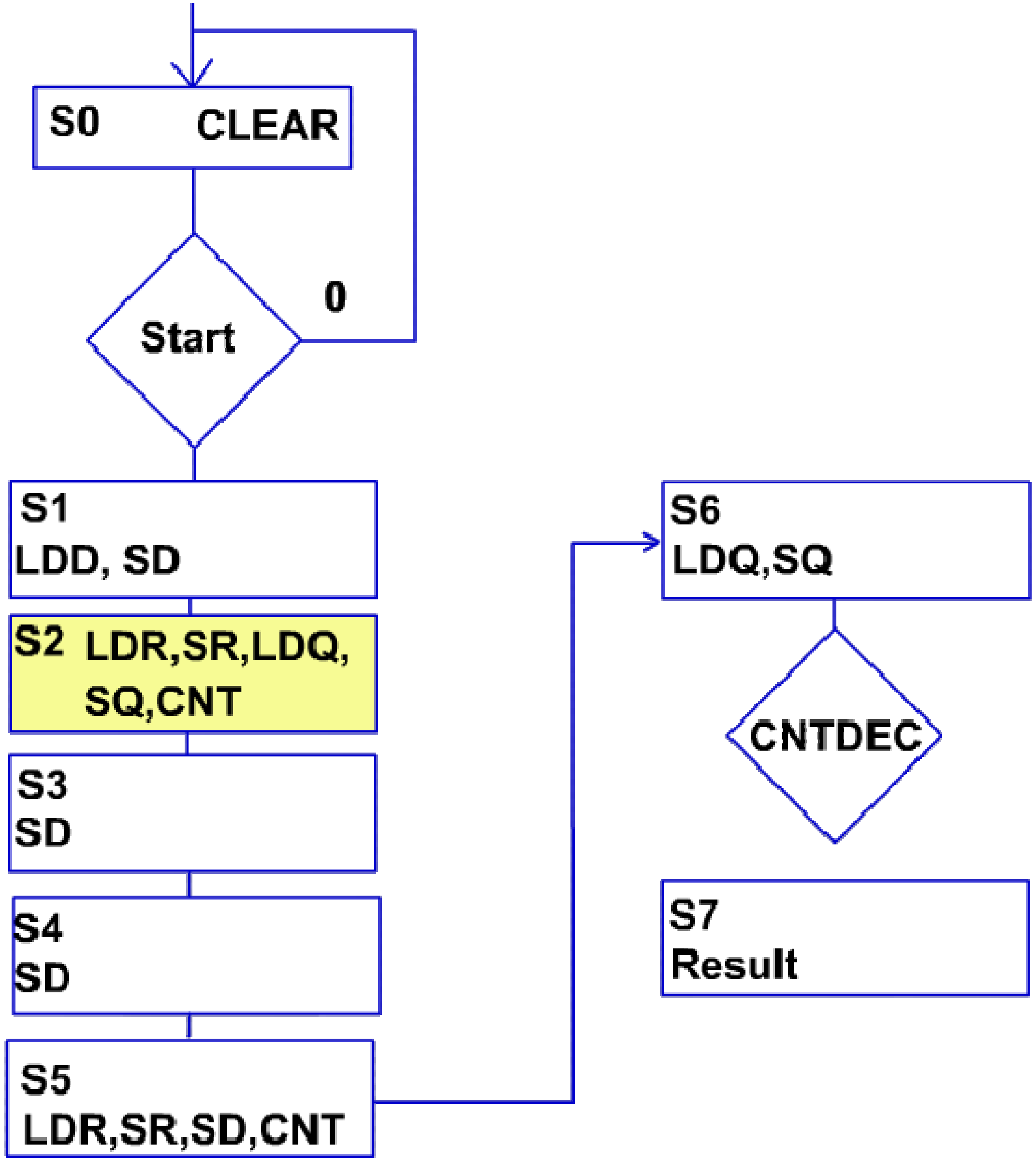
Square Root Operation

ASM



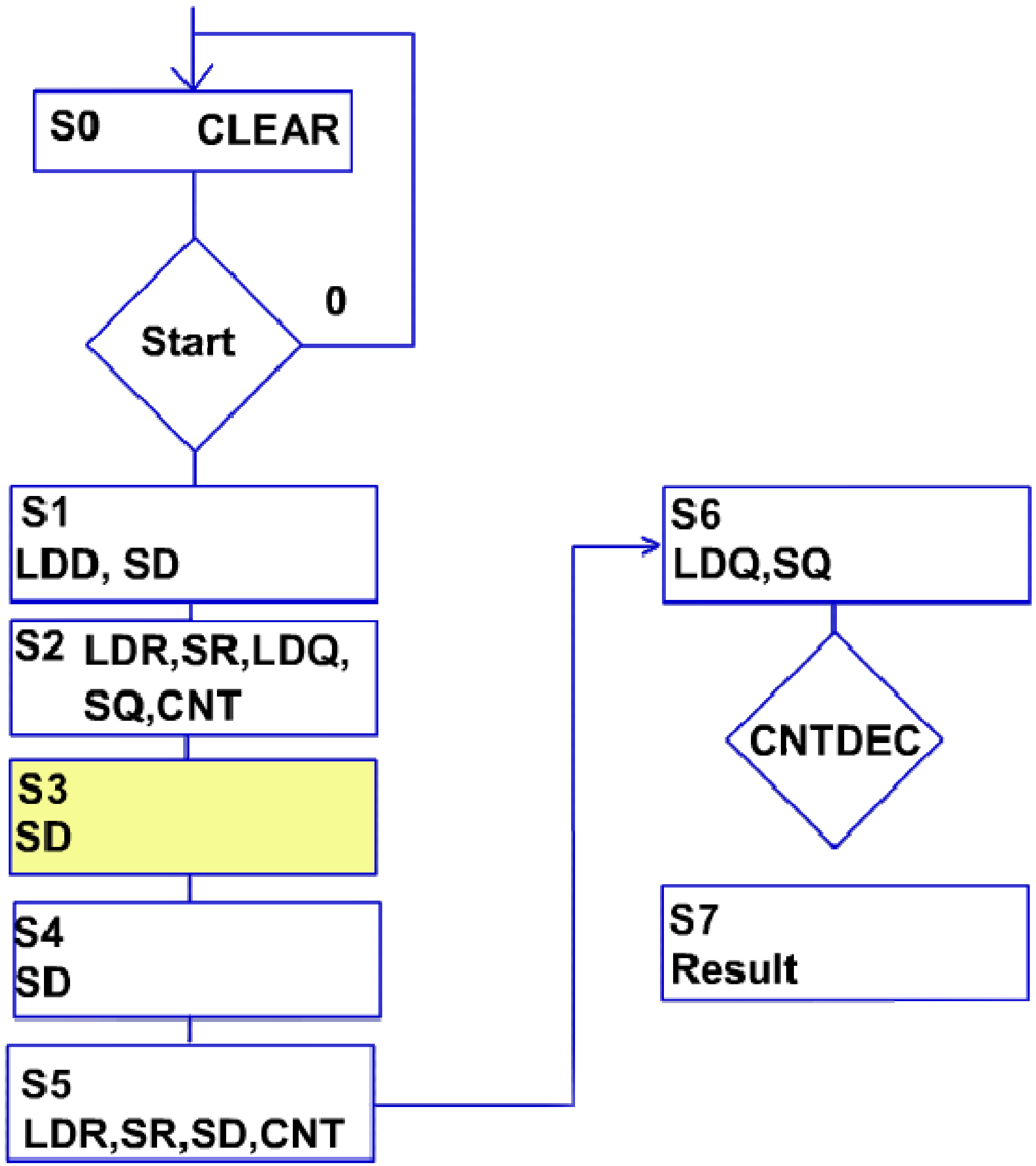
Square Root Operation

ASM



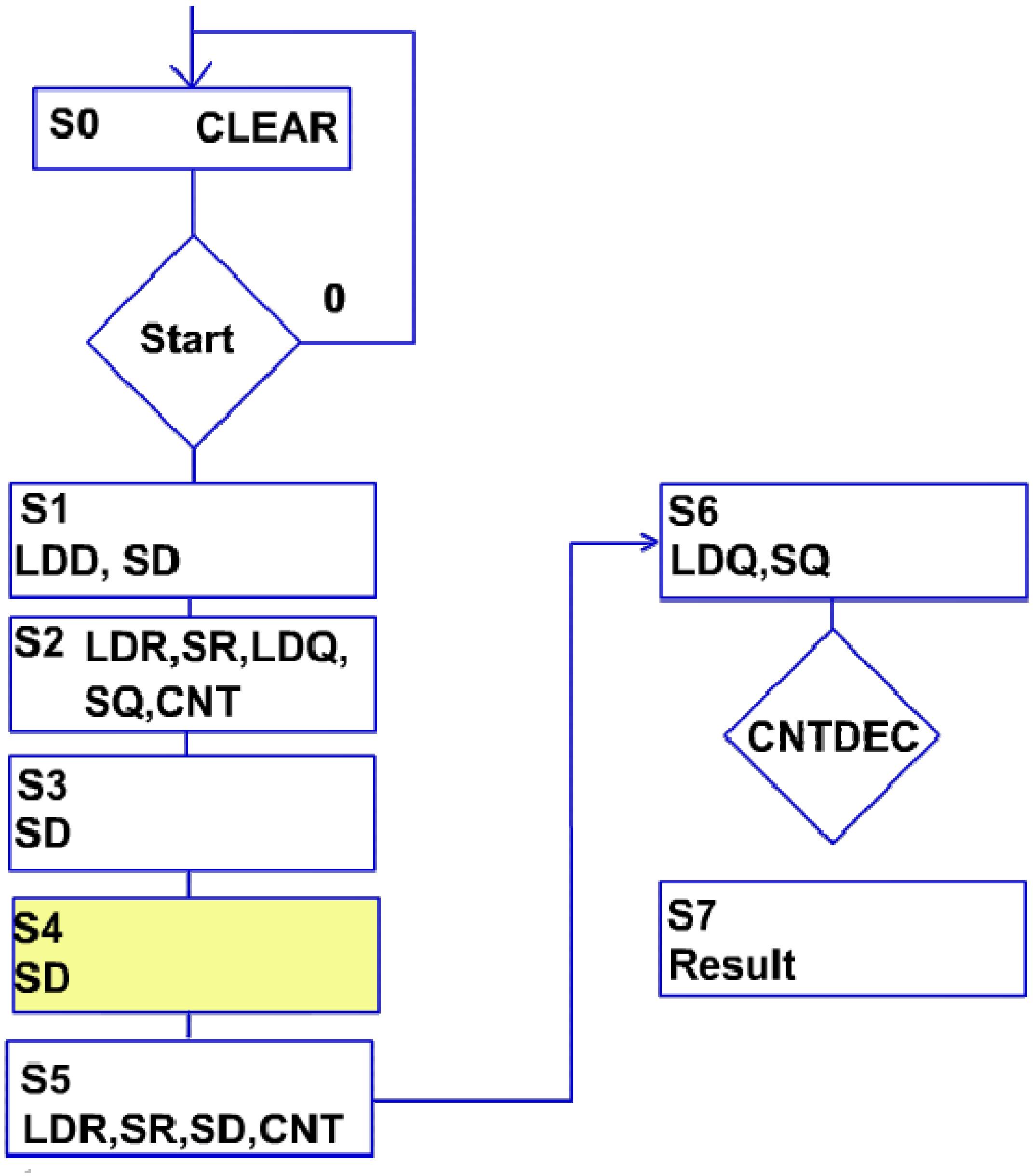
Square Root Operation

ASM



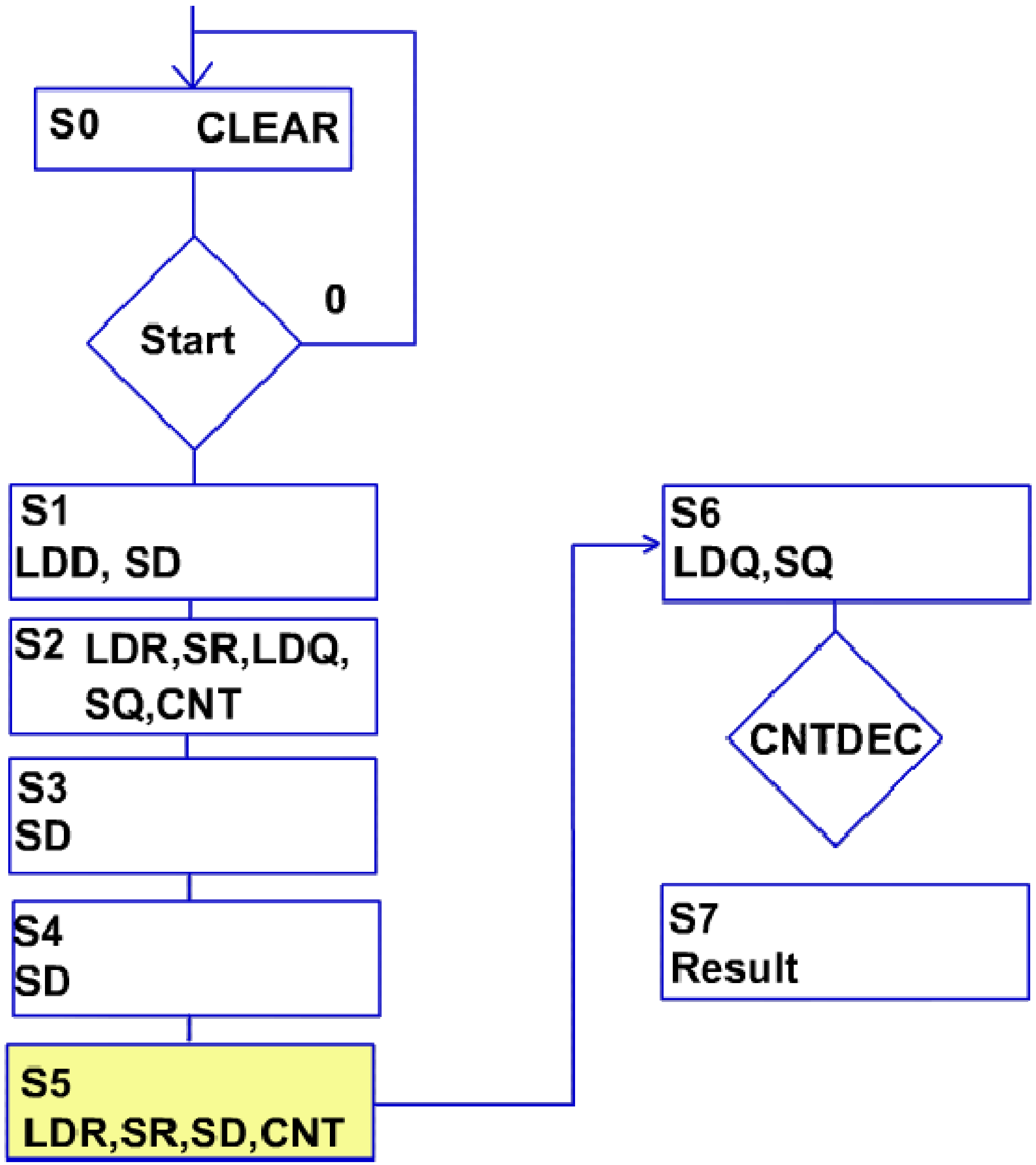
Square Root Operation

ASM



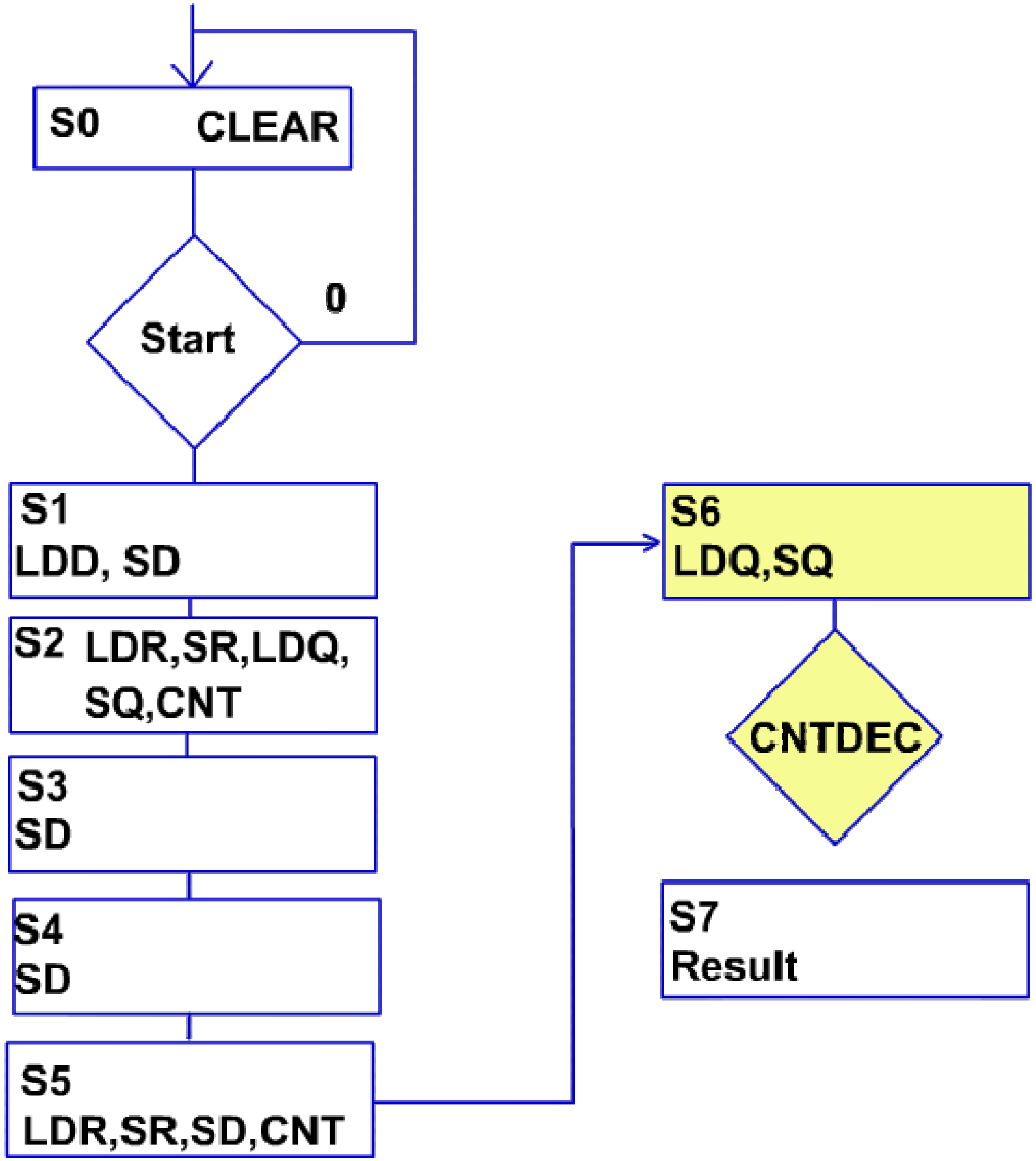
Square Root Operation

ASM



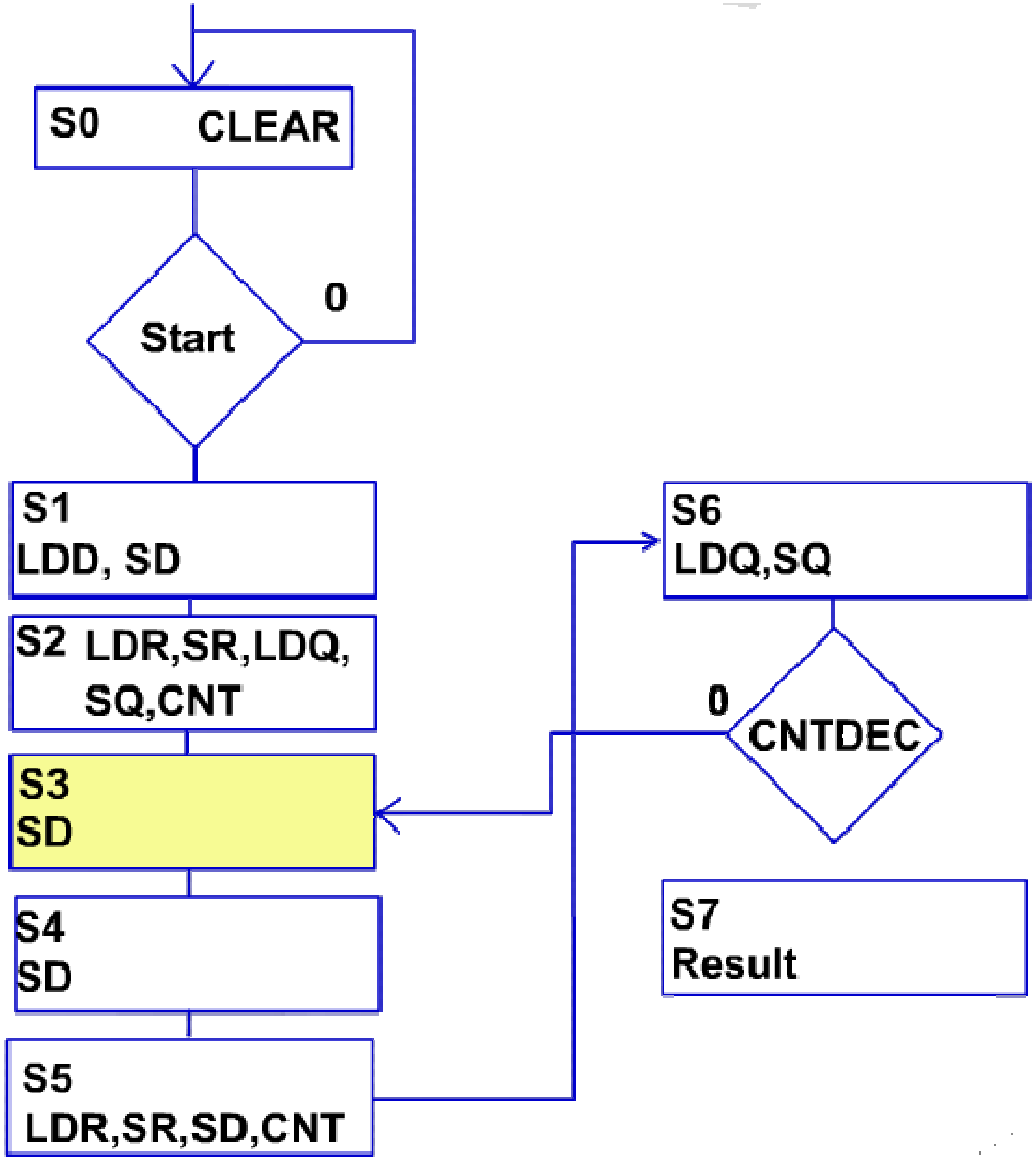
Square Root Operation

ASM



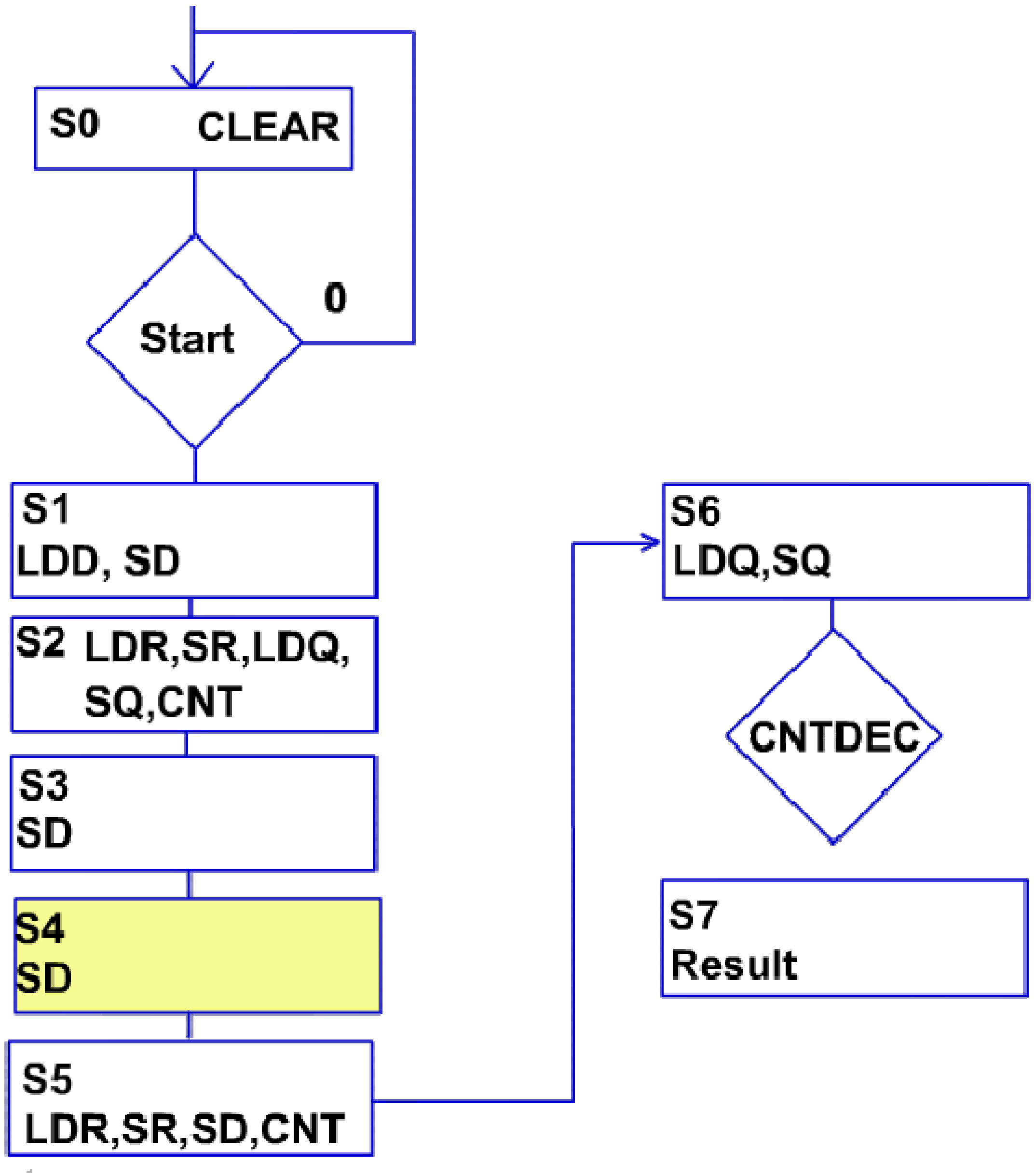
Square Root Operation

ASM



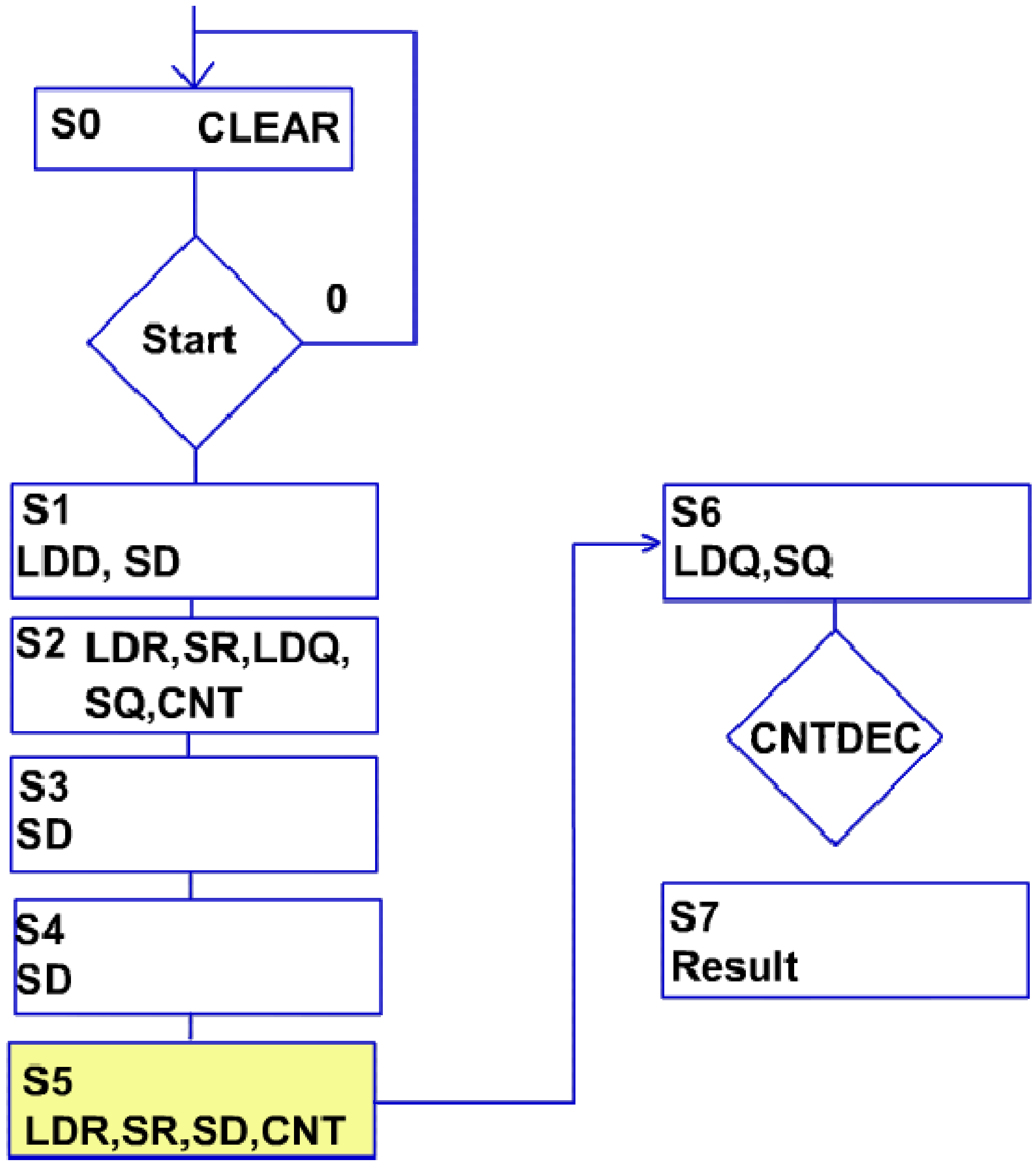
Square Root Operation

ASM



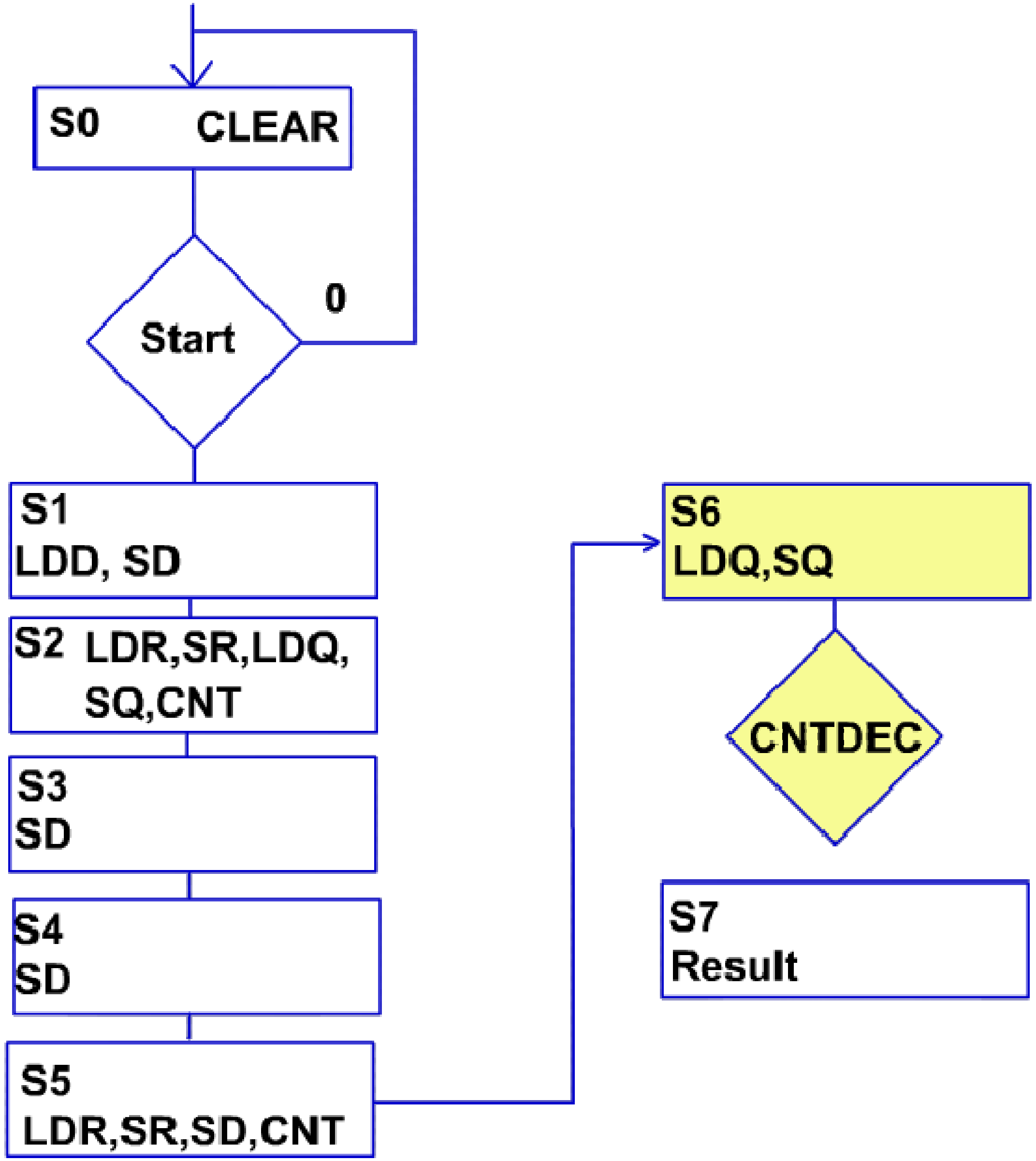
Square Root Operation

ASM



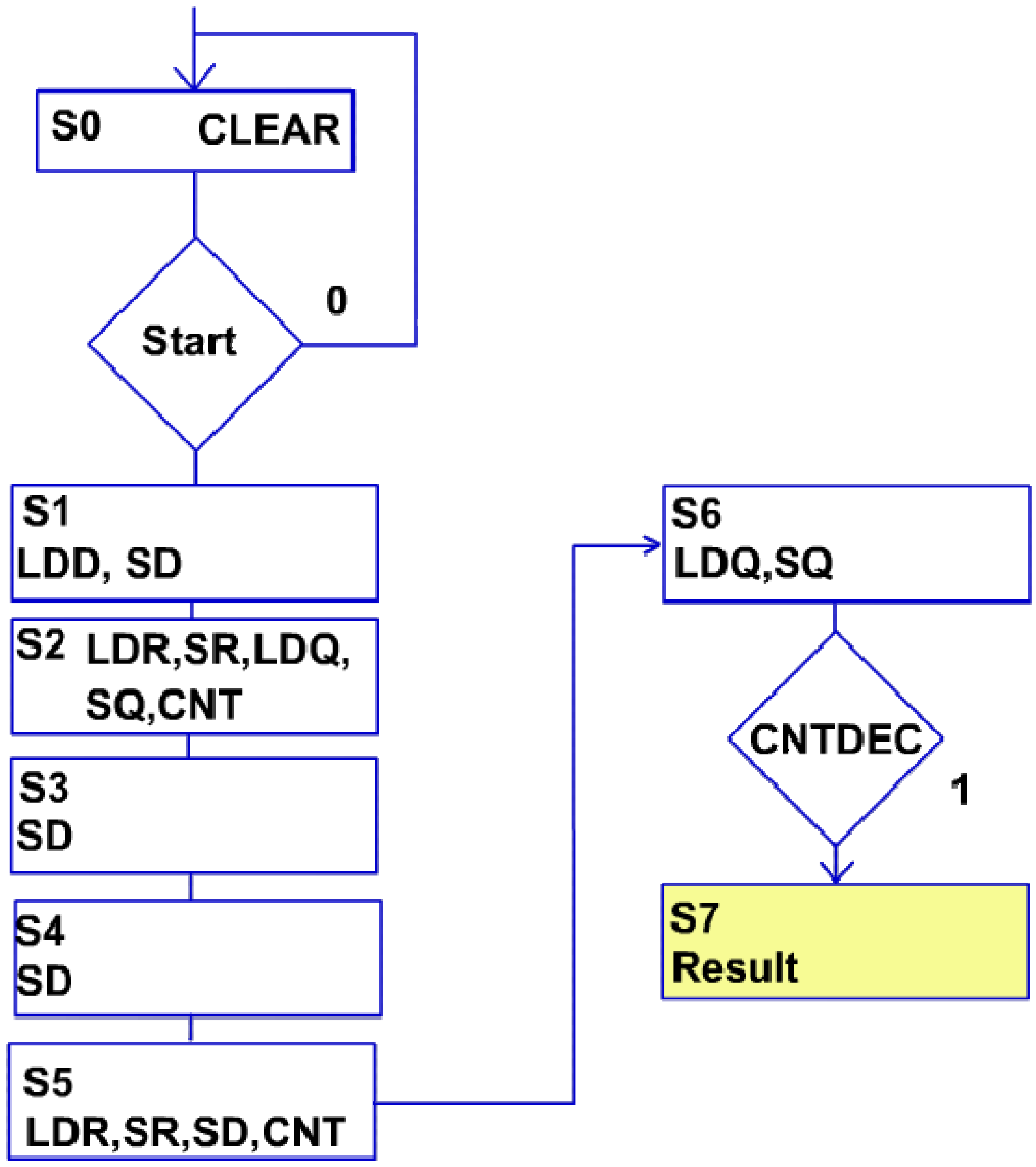
Square Root Operation

ASM



Square Root Operation

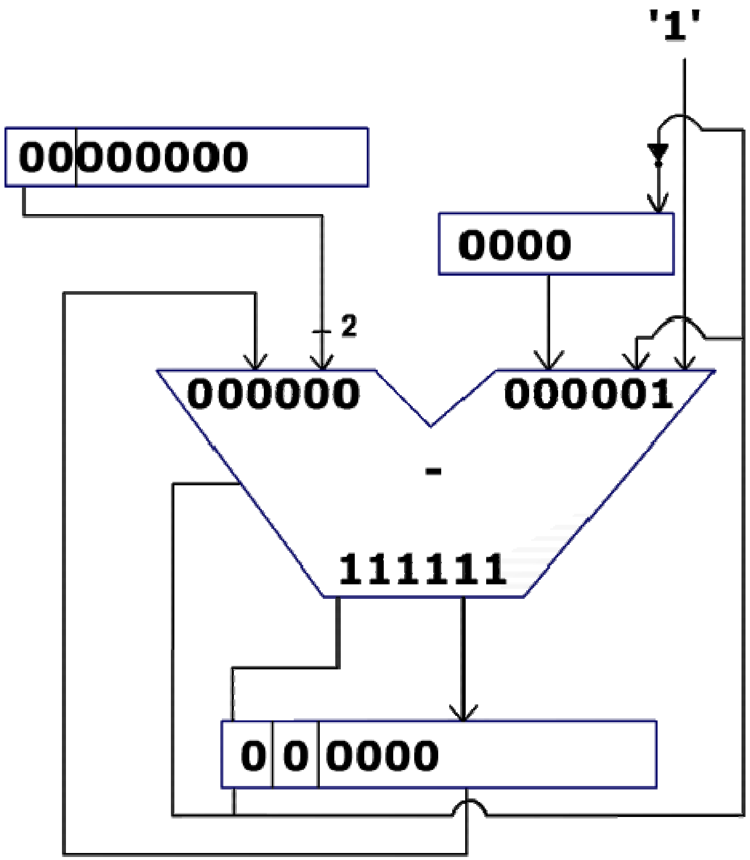
ASM



Square Root Operation

Example

The Radincand is a number of 8 bits 140 (10001100_2), the solution Q should be 11 (1011_2), and the remainder R should be 19 (10011_2).

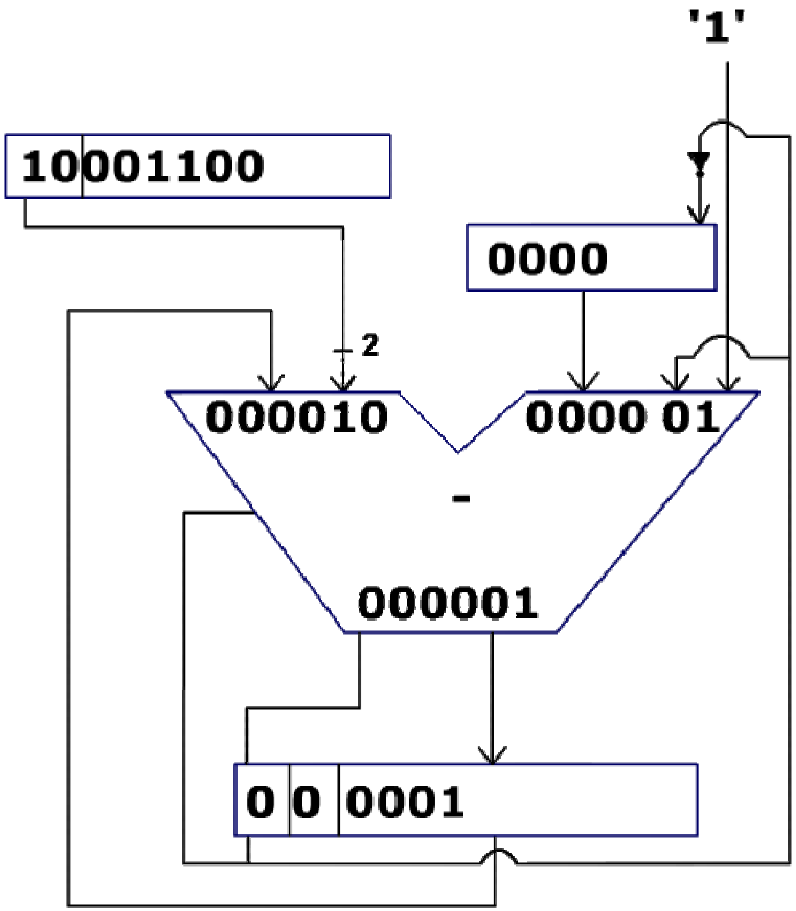


Start set R = 000000 and Q = 0000

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (100011002), the solution Q should be 11 (10112), and the remainder R should be 19 (100112).



D = 10001100

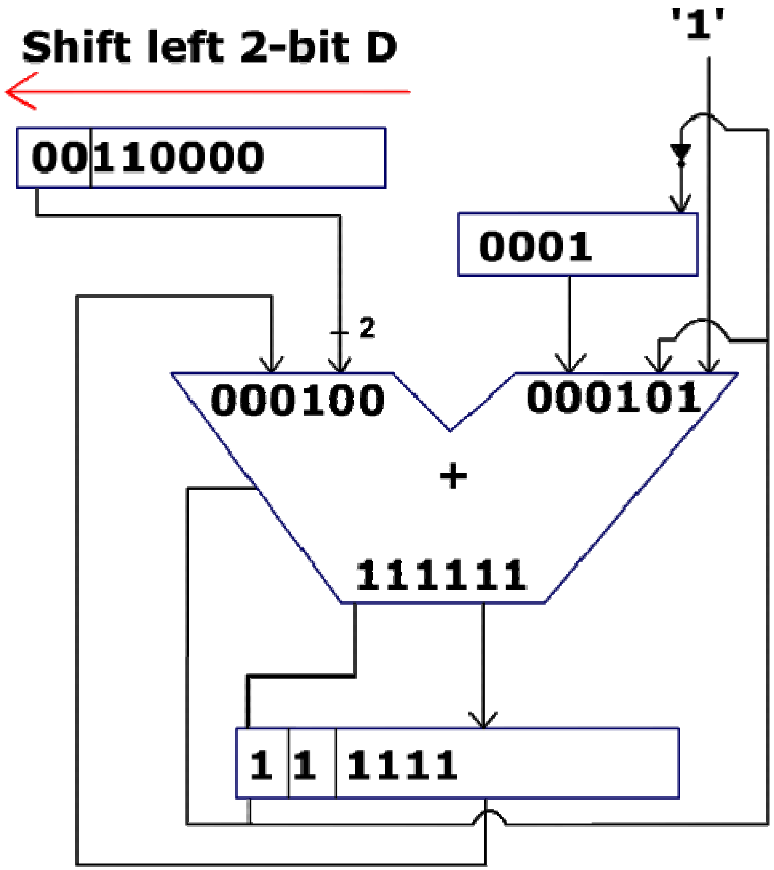
R = R or (D & 192)

R = R - (Q or 1)

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (100011002), the solution Q should be 11 (10112), and the remainder R should be 19 (100112).



Q = (Q or 1)

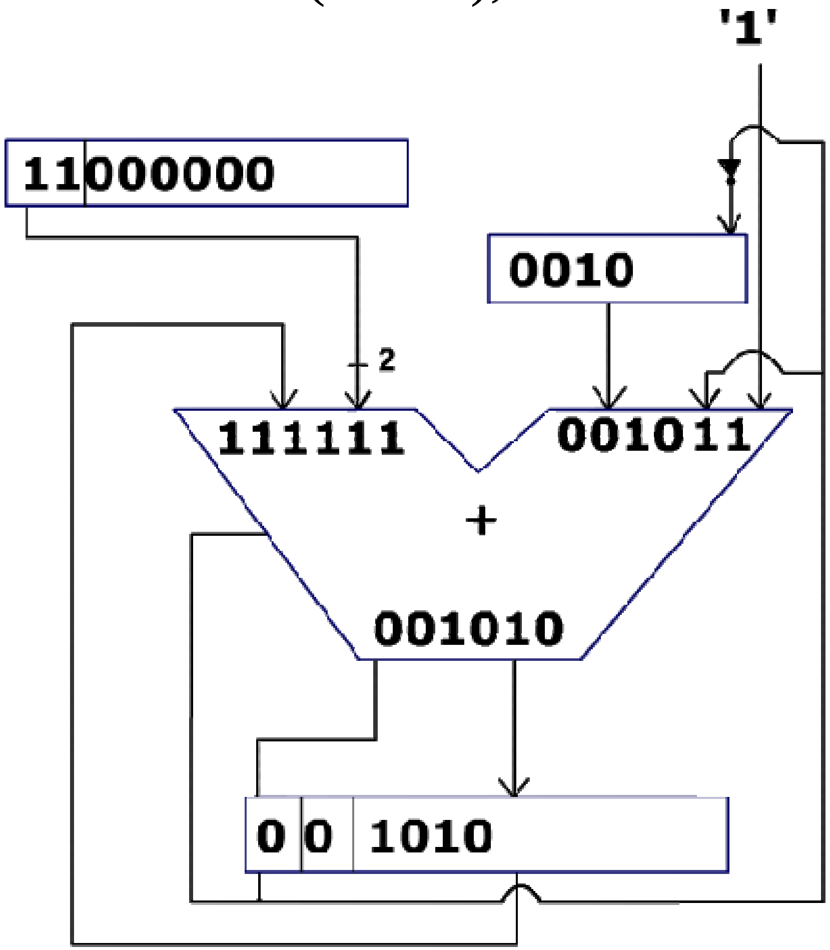
R = (R <<2) or ((D << 2) & 192)

R = R - (Q or 1)

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (100011002), the solution Q should be 11 (10112), and the remainder R should be 19 (100112).



Q = Q or 0

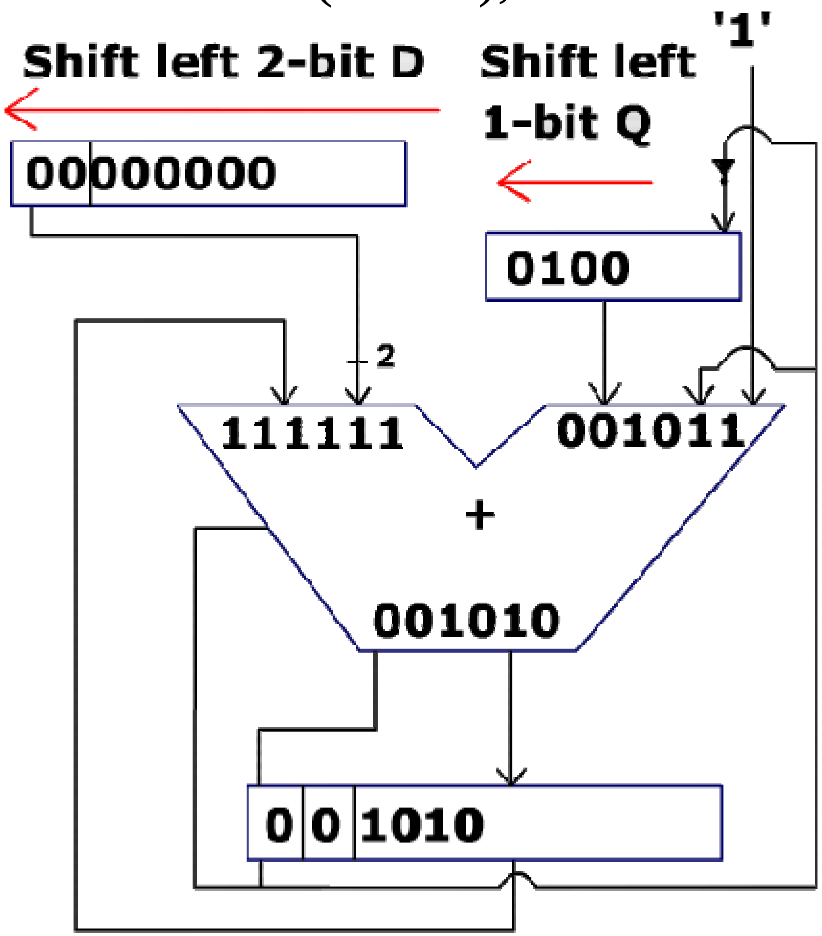
R = (R <<2) or ((D <<2) & 192)

R = R + (Q or 3)

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (100011002), the solution Q should be 11 (10112), and the remainder R should be 19 (100112).



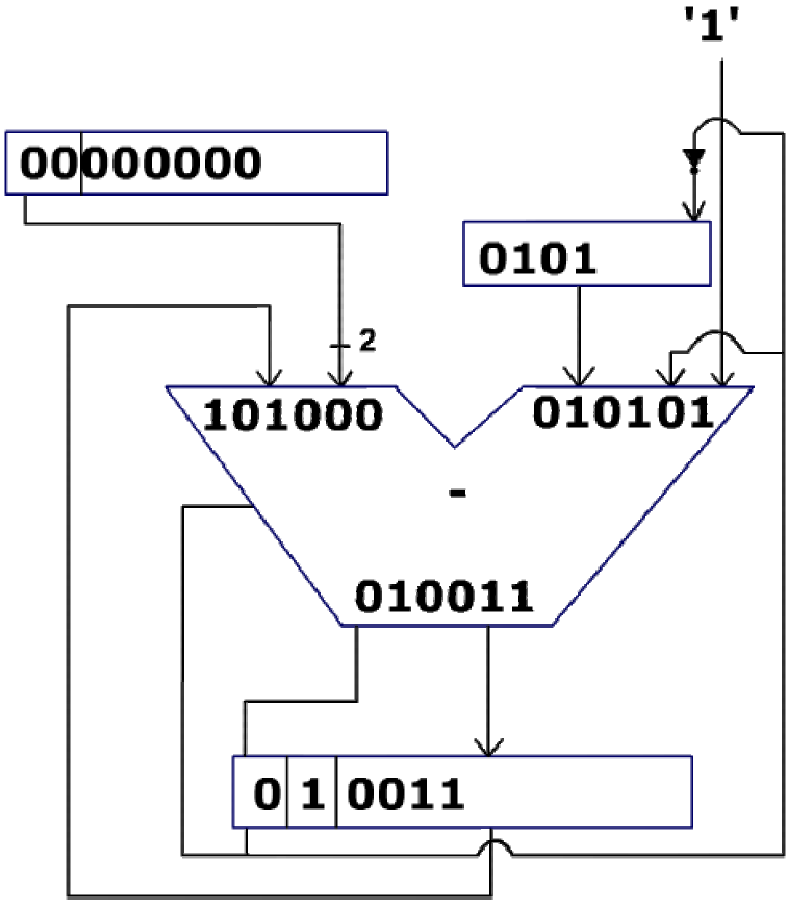
$D = D \ll 2$

$Q = Q \ll 1$

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (100011002), the solution Q should be 11 (10112), and the remainder R should be 19 (100112).



Q = Q or 1

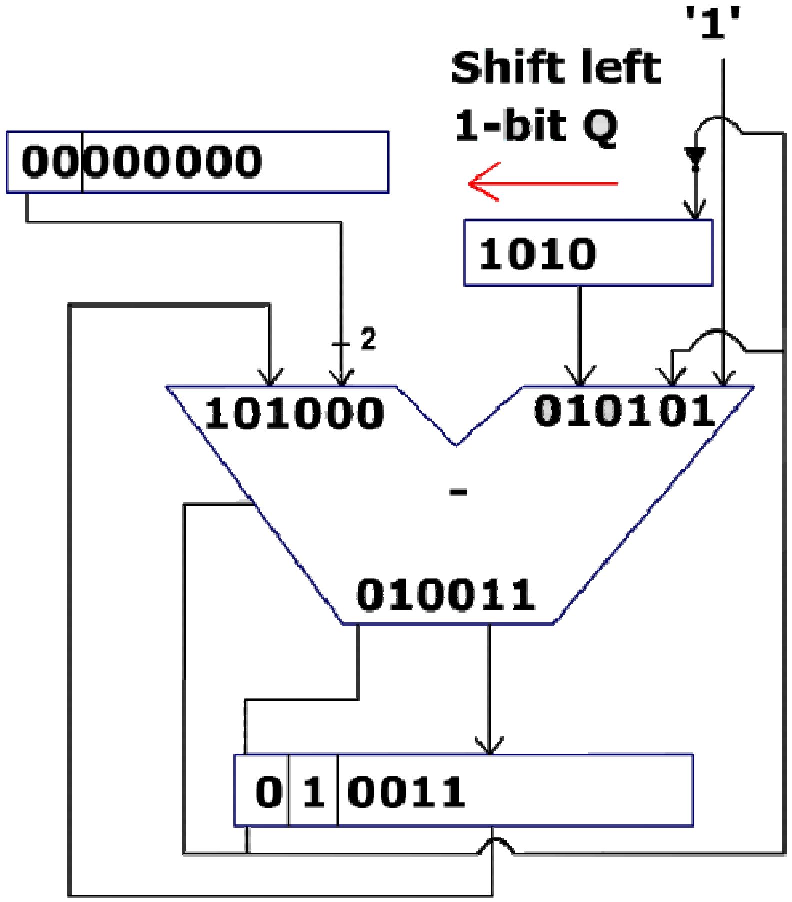
R = (R <<2) or ((D << 2) & 192)

R = R - (Q or 1)

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (10001100_2), the solution Q should be 11 (1011_2), and the remainder R should be 19 (10011_2).

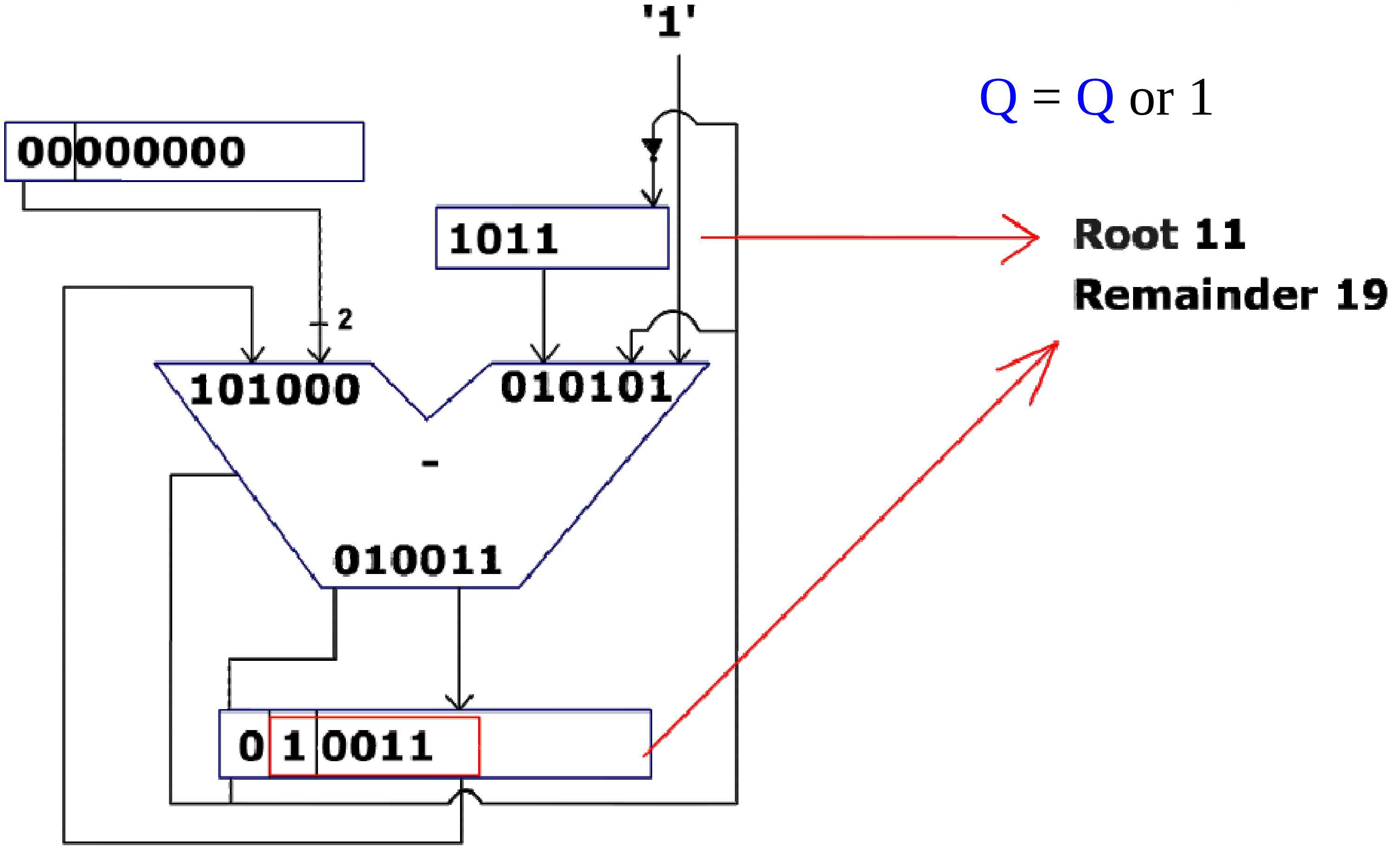


Q = Q << 1

Square Root Operation

Example

The Radincand is a number of 8 bits 140 (100011002), the solution Q should be 11 (10112), and the remainder R should be 19 (100112).



Square Root Operation

The Circuit

Using MAX-PLUS design tools , the circuit was test.

To this circuit was used the follow devices:

74198 8-bit Shift Register

74194 4-bit Shift Register

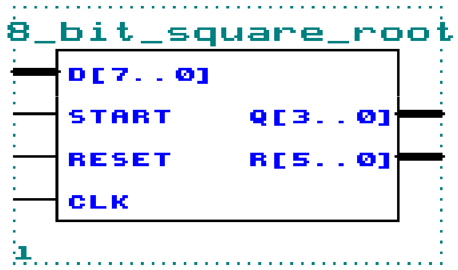
6-bit Register maked with megafunction

Add-Sub maked with megafunction

4-bit Counter down maked with megafunction

Fsm maked in AHDL

Some gates



Square Root Operation

The FSM

```
SUBDESIGN fsm
(
   clk, reset   : INPUT;
   start, cnt   : INPUT;
   LDD, SD, LDR, SR, LDQ, SQ, CLEAR, CNTD  : OUTPUT
)
VARIABLE

   ss: MACHINE OF BITS (LDD, SD, LDR, SR, LDQ, SQ, CLEAR, CNTD)

      WITH STATES  (
         s0  = B"00000000",
         s1  = B"11000010",
         s2  = B"00111111",
         s3  = B"01000010",
         s4  = B"01000010",
         s5  = B"00110111",
         s6  = B"00001110",
         s7  = B"00000010");
```

Square Root Operation

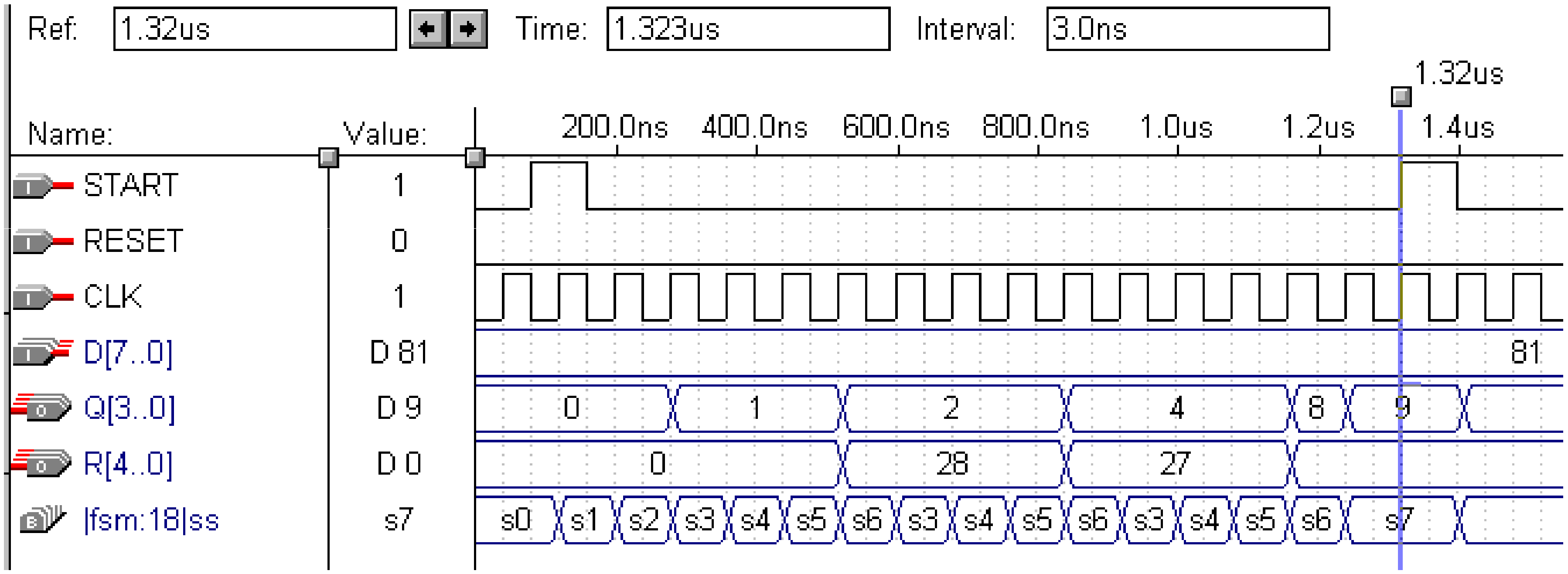
The FSM

```
BEGIN
   ss.clk   = clk;
   ss.reset = reset;

   TABLE
      ss,   start,   cnt =>   ss;
      s0,   0,        x    =>   s0;
      s0,   1,        x    =>   s1;
      s1,   x,        x    =>   s2;
      s2,   x,        x    =>   s3;
      s3,   x,        x    =>   s4;
      s4,   x,        x    =>   s5;
      s5,   x,        x    =>   s6;
      s6,   x,        0    =>   s3;
      s6,   x,        1    =>   s7;
      s7,   0,        1    =>   s7;
      s7,   1,        1    =>   s0;
   END TABLE;
END;
```

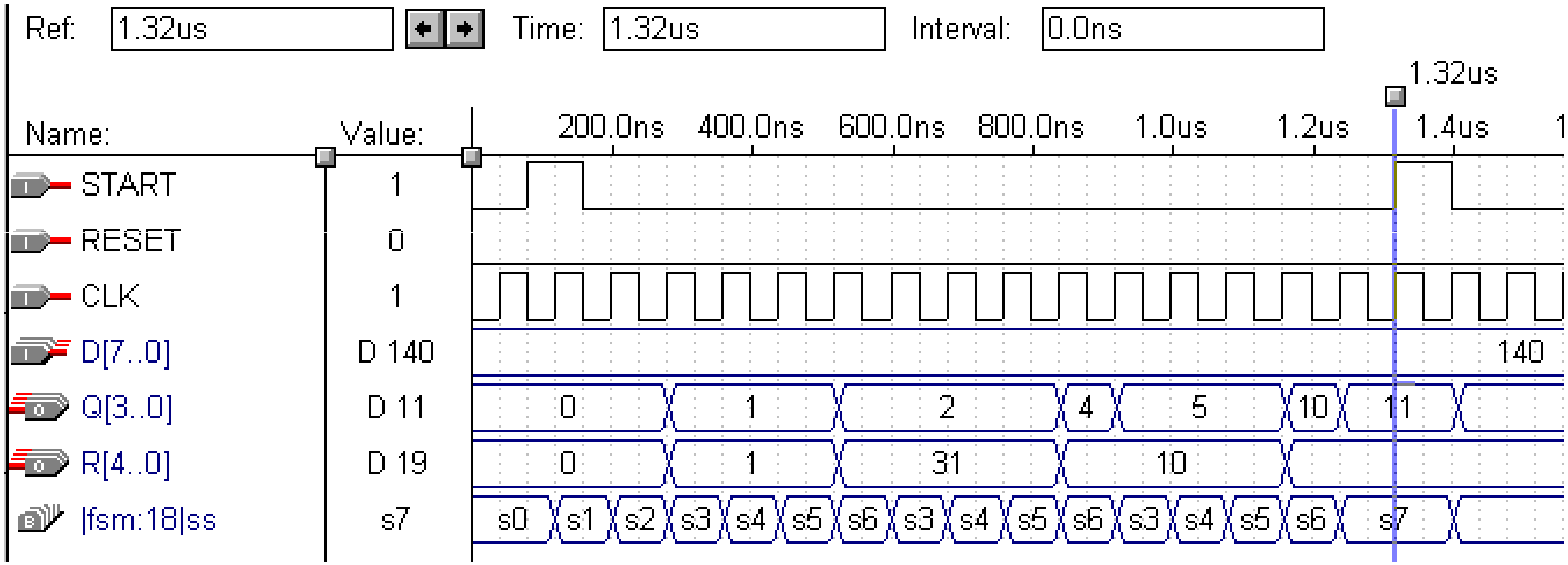
Square Root Operation

Example 81



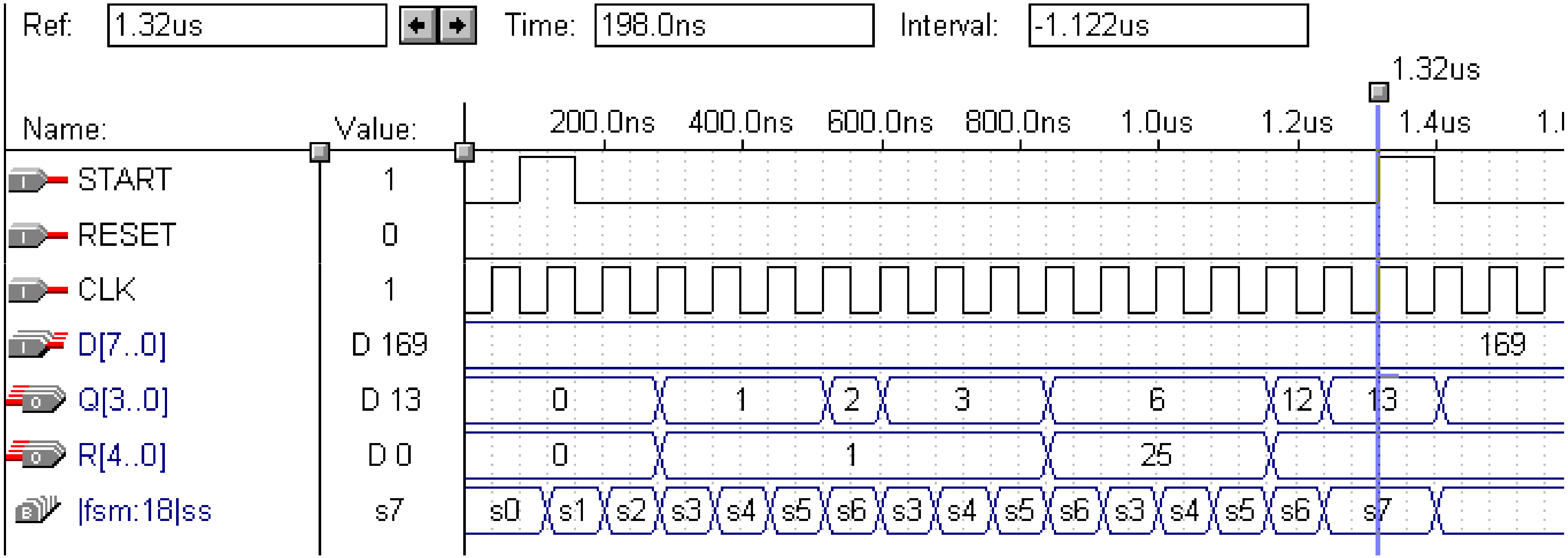
Square Root Operation

Example 140



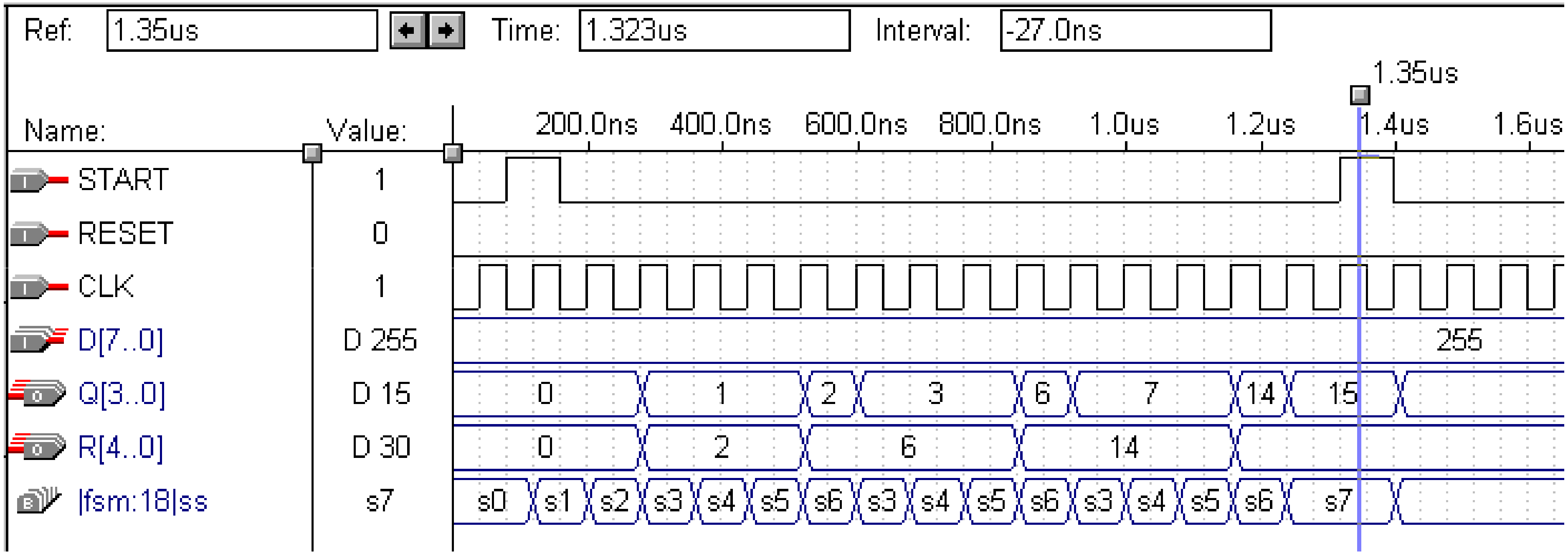
Square Root Operation

Example 169



Square Root Operation

Example 255



Square Root Operation

References

An FPGA Implementation on a Fixed-Point Square Root Operation

K. Piromsopa, C. Aporntewan and P. Chongsatitvatana