EE 486 lecture 12: More on Divide, systems issues and SRT.

M. J. Flynn

(Figures used in slides 4-13 are from B. Parhami, Computer Arithmetic)

Non restoring division

- We define the partial remainder as:
  $s^{(j)} = r\, s^{(j-1)} - q_j\, d$
- For binary, r =2 and q_j in{-1,1}
- So we end up with something like:
  1-1-1111-1

Conversion to binary

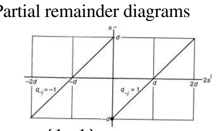
- Shift left by 1
- Pad 1 as the new LSB
- Keep the 1 as is and replace any -1 by 0
- Complement the MSB

Proof: use $b_i = (q_i+1)/2$ or $q_i = 2\, b_i - 1$

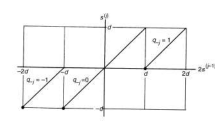
Non restoring; two views

Partial remainder diagrams

- These are 2 Partial remainder diagrams;
  - 2 possible quotient digits (or actions)
  - 3 digits; now including the possibility of a 0 quotient digit (or no op action).
  - The 3 digit set allows a redundant representation



{1, -1}



{1,0,-1}

Partial Remainder diagrams

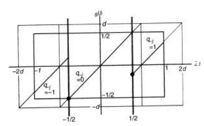
- Show the partial remainder's range and the quotient digit to be selected.
- Using the digit set {-1,1} recognizes non restore correction but each iteration has a full CPA delay.
- Using {-1,0,1} allows us to recognize the skip over 0 case and do a no op (Software, variable shift). Also, redundancy allows the delay of only a CSA per iteration (Hardware).

SRT

- Sweeney, Robertson and Tocher (SRT)
- Use digit redundancy to simplify/ speed up divide.
- If we have a redundant set {-1,0,1} for some combination of $s^{(j)}$ and d (PD combination) we can select either 0 or –1: or 0 or 1 and still get the same result.

Radix-2 SRT

- Choose q to be 0 in [-1/2, +1/2) and 1 for P>= +1/2 and –1 for P <-1/2
- Now comparisons are with =+/-1/2 not with D.



Radix-2 SRT

- If P is in [-1/2,+1/2) then q = 0; skip and shift.
- If P >= +1/2 then q = 1 and subtract D
- If P < -1/2 then q = -1 and add D
- Correct by subtracting -1s from 0s

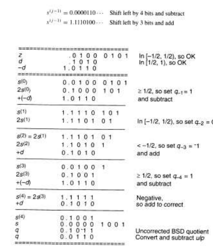
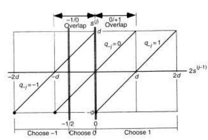


Fig. 14.6 Example of unsigned radix-2 SRT division.

Selecting regions

- Overlapping regions can use either digit for quotient. Want to make the selection to be easy and well as speeding up the divisor add/subtract.



Using CSAs

- By forming only a high 4b sum for P, we can select the correct action; the rest of P can be left in S+C form and returned to the next iteration thru a CSA (combined with the D multiple)

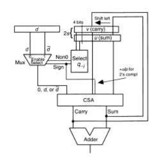
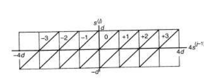


Fig. 14.8 Block diagram of a radix-2 divider with partial remainder in stored-carry form.

Radix-4 SRT

- Clearly, we want more than 1b per iteration
- Radix-4 gives 2b per iteration.
- There's a big tradeoff between the redundancy in the quotient digit set and the quotient selection complexity.
- {-3,-2,-1,0,1,2,3} provides easy selection (lots of overlap) but requires +/- 3D.
- {-2,-1,0,1,2} makes selection more complex

Radix-4 SRT



Using {-3,…,+3)
Note the overlap above
And the easy selection (right)

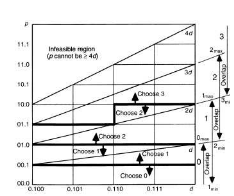
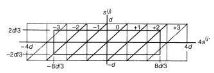
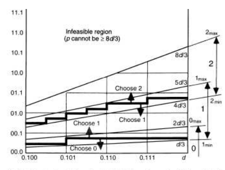


Fig. 14.12 A p-d plot for radix-4 SRT division with quotient digit set [−3, 3]

Radix-4 SRT





Digit set {-2,..+2}now easy generation but the selection (right) is more complex.

SRT

- Has been the most widely used (especially radix-4); radix-8 is also sometimes used. That's probably the practical limit though it's possible to pipeline 2 lower order SRT to get the equivalent of a higher order SRT.
- Since it's subtractive, SRT gives IEEE quotient and the remainder.