DIGIT-SERIAL ARITHMETIC

- Modes of operation:LSDF and MSDF
- Algorithm and implementation models
- LSDF arithmetic
- MSDF: Online arithmetic

TIMING PARAMETERS

- radix-r number system: conventional and redundant
- Serial signal: a numerical input or output with one digit per clock cycle
- For an n digit signal, the clock cycles are numbered from 1 to n
- Timing characteristics of a serial operation determined by two parameters:
  - *initial delay* δ: additional number of operand digits required to determine the first result digit
  - *execution time* T_n; the time between the first digit of the operand and the last digit of the result

$$T_n = \delta + n + 1$$

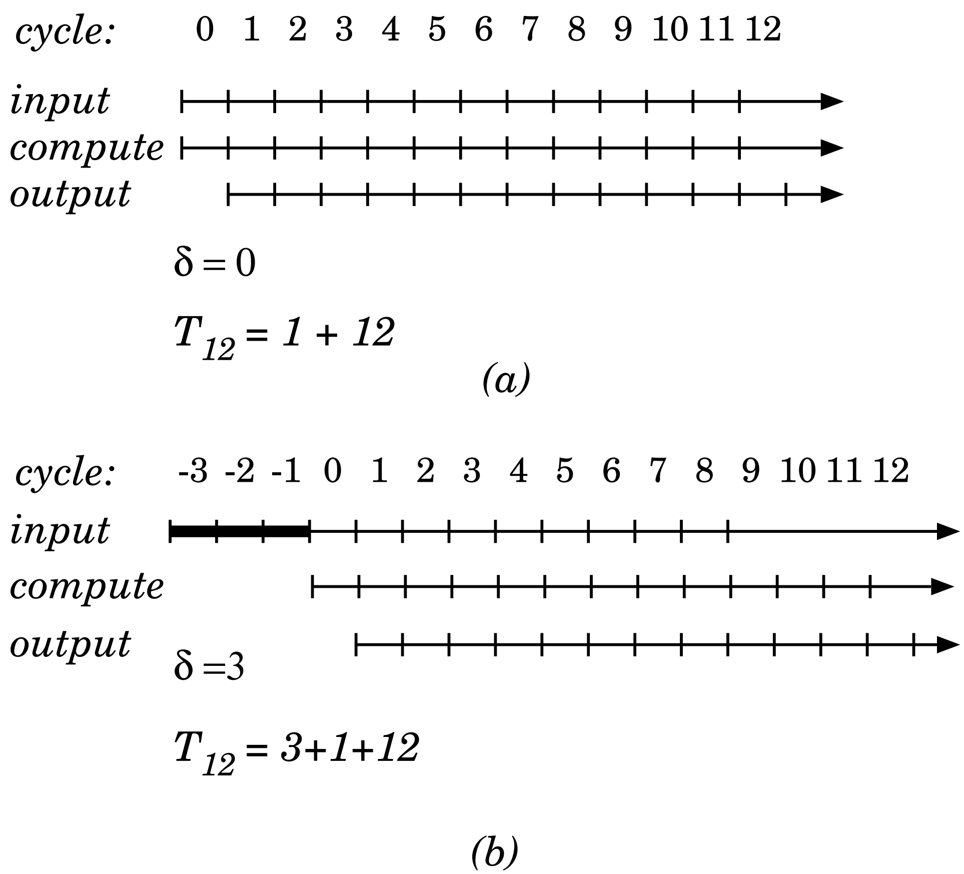


Figure 9.1: Timing characteristics of serial operation with $n = 12$. (a) With $\delta = 0$. (b) With $\delta = 3$.

LSDF AND MSDF

1. *Least-significant digit first* (LSDF) mode (right-to-left mode)

$$x = \sum_{i=0}^{n-1} x_i r^i$$

2. *Most-significant digit first* (MSDF) mode (left-to-right mode)

   also known as *online arithmetic*

   initial delay called *online delay*

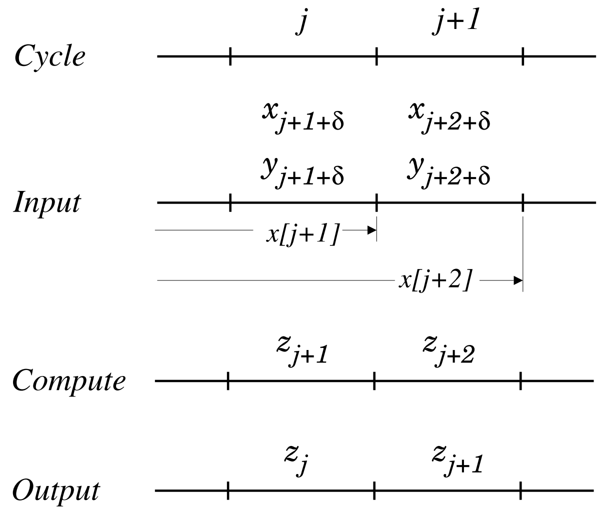
$$x = \sum_{i=1}^{n} x_i r^{-i}$$

ALGORITHM MODEL

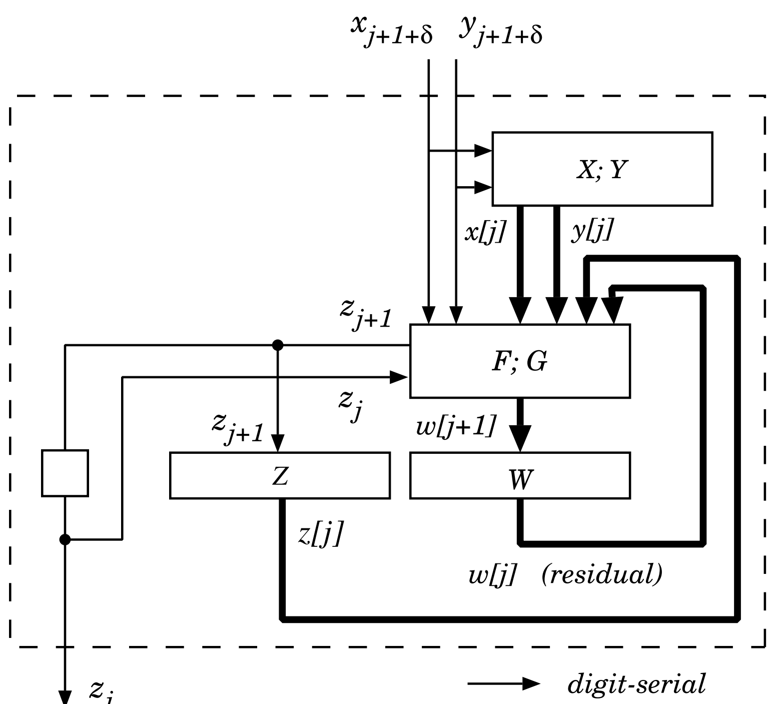
- Operands x and y, result z: n radix-r digits
- In cycle j the result digit z_{j+1} is computed
- Cycles labeled $-\delta, \ldots, 0, 1, \ldots, n$
- In cycle j receive the operand digits $x_{j+1+\delta}$ and $y_{j+1+\delta}$, and output z_j
- $x[j]$, $y[j]$ and $z[j]$ are the numerical values of the corresponding signals when their representation consists of the first $j+\delta$ digits for the operands, and j digits for the result.

In iteration j

$$
\begin{aligned}
x[j+1] &= (x[j], x_{j+1+\delta}) \\
y[j+1] &= (y[j], y_{j+1+\delta}) \\
z_{j+1} &= F(w[j], x[j], x_{j+1+\delta}, y[j], y_{j+1+\delta}, z[j]) \\
z[j+1] &= (z[j,], z_{j+1}) \\
w[j+1] &= G(w[j], x[j], x_{j+1+\delta}, y[j], y_{j+1+\delta}, z[j], z_{j+1})
\end{aligned}
$$


j
j+1
Cycle
$x_{j+1+\delta}$
$x_{j+2+\delta}$
$y_{j+1+\delta}$
$y_{j+2+\delta}$
Input
x[j+1]
x[j+2]
z_{j+1}
z_{j+2}
Compute
z_j
z_{j+1}
Output

(a)


$x_{j+1+\delta}$
$y_{j+1+\delta}$
X; Y
x[j]
y[j]
z_{j+1}
F; G
z_j
z_{j+1}
w[j+1]
Z
W
z[j]
w[j] (residual)
z_j
digit-serial
digit-parallel

(b)

Table 9.1: Initial delay (δ)

Operation	LSDF	MSDF
Addition	0	2 ($r = 2$)
		1 ($r \geq 4$)
Multiplication	0	3 ($r = 2$)
		2 ($r = 4$)
(only MS half)	n	
Division	$2n$	4
Square root	$2n$	4
Max/min	n	0

```
a) LSDF mode

n-digit addition:
  Cycle:     0 1 2 . . .
            LSD             MSD
  Inputs:    x x x x x x x x x
  Output:      x x x x x x x x x

n by n --> 2n multiplication:

            LSD             MSD
  Inputs:    x x x x x x x x x
  Output:      x x x x x x x x x x x x x x x x x x
                                 -----------------
                                      MS half

b) MSDF mode

  Cycle:    -2 -1 0 1 2 . . .
n-digit operation:
            MSD                   LSD
  Inputs:    x  x x x x  x  x  x  x
  Output:           x x  x  x  x  x  x  x  x
             ---
          online delay = 2
```

Figure 9.3: LSDF and MSDF modes.

COMPOSITE ALGORITHM: EXAMPLE

- Givens method for matrix triangularization
- Rotation factors:

$$c = \frac{x}{\sqrt{x^2 + y^2}} \quad s = \frac{y}{\sqrt{x^2 + y^2}}$$

- The online delay of the network

$$\Delta_{rot} = \delta 1 + \delta 2 + \delta 3 + \delta 4 = 13$$

- Execution time (latency): $D_{rot} = \Delta_{rot} + n + 4$

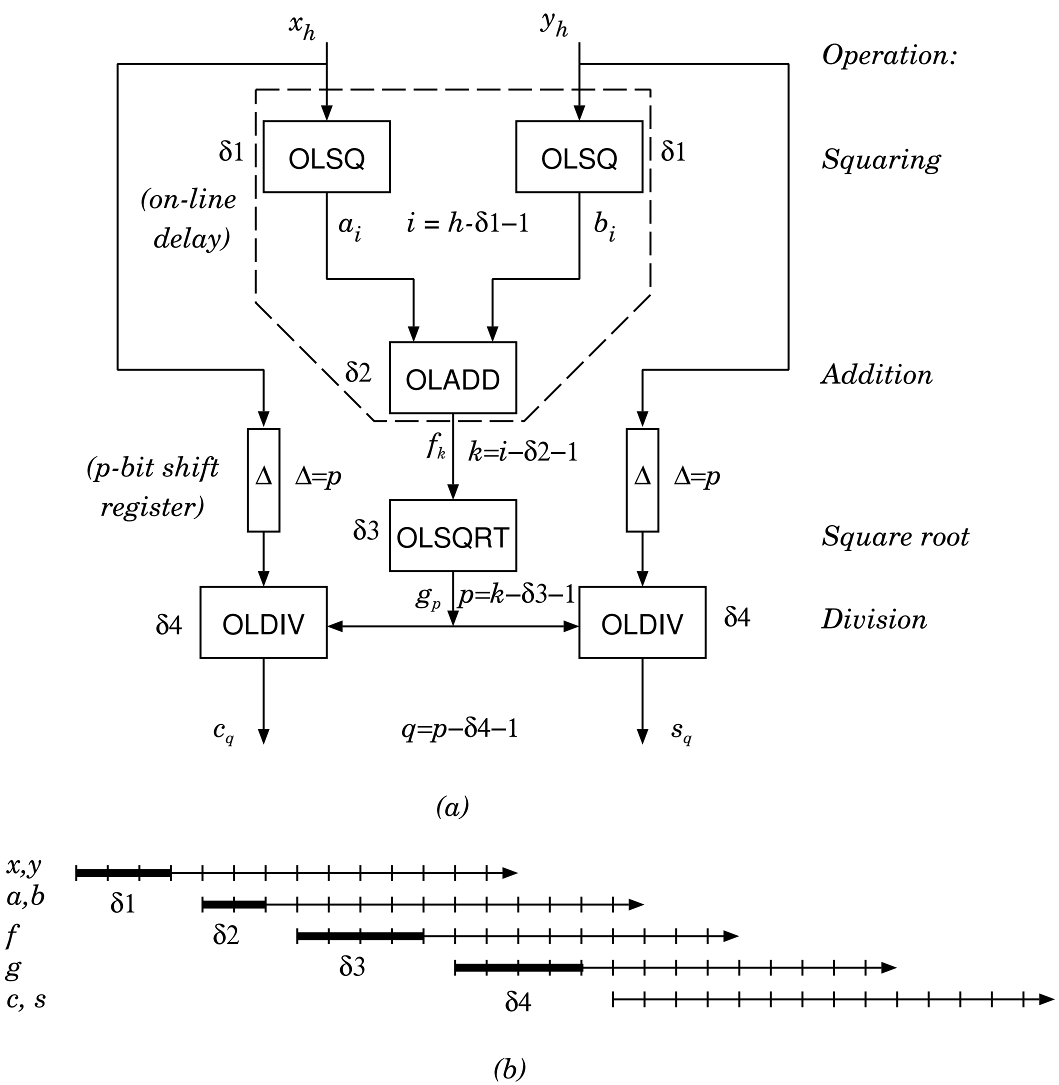


Figure 9.4: Online computation of rotation factors: (a) Network. (b) Timing diagram.

LSDF ARITHMETIC: ADDITION/SUBTRACTION

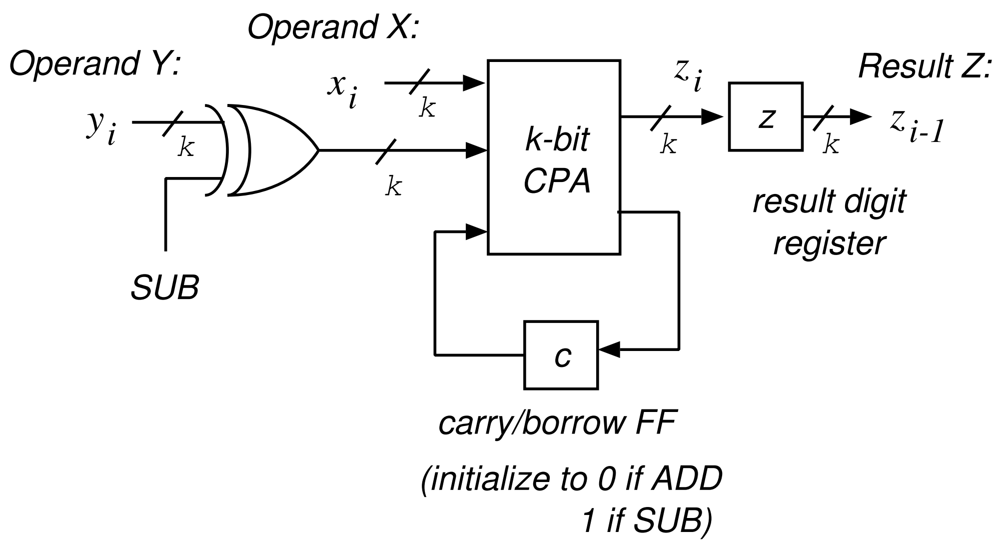
- The cycle delay

$$t_{LSDFadd-k} = t_{CPA(k)} + t_{FF}$$

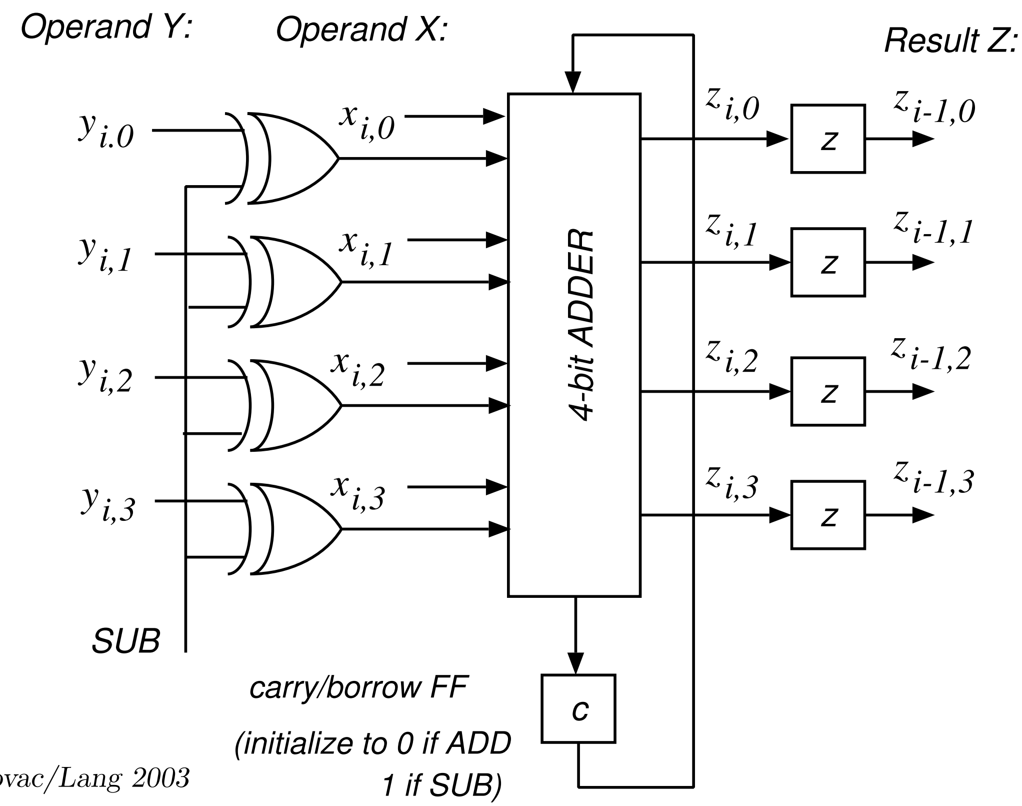
- The total time for n-bit addition

$$T_{LSDFadd-n} = (\frac{n}{k} + 1)t_{LSDFadd-k}$$

- The cost: one k-bit CPA, one flip-flop, and one k-bit output register


Operand X:
Operand Y:
x_i
k
y_i
k
k
k-bit
CPA
z_i
k
z
k
Result Z:
z_{i-1}
result digit
register
SUB
c
carry/borrow FF
(initialize to 0 if ADD
1 if SUB)

(a)


Operand Y:
Operand X:
Result Z:
$y_{i.0}$
$x_{i,0}$
$y_{i,1}$
$x_{i,1}$
$y_{i,2}$
$x_{i,2}$
$y_{i,3}$
$x_{i,3}$
4-bit ADDER
$z_{i,0}$
$z_{i,1}$
$z_{i,2}$
$z_{i,3}$
z
z
z
z
$z_{i-1,0}$
$z_{i-1,1}$
$z_{i-1,2}$
$z_{i-1,3}$
SUB
c
carry/borrow FF
(initialize to 0 if ADD
1 if SUB)

LSDF MULTIPLICATION

- For radix-2 and 2's complement representation:

1. Serial-serial (LSDF-SS) multiplier, both operands used in digit-serial form.
2. Serial-parallel (LSDF-SP) multiplier, one operand first converted to parallel form

- Operation cannot be completed during the input of operands

SERIAL-SERIAL MULTIPLIER

- Define internal state (residual)

$$w[j] = 2^{-(j+1)}(x[j] \times y[j] - p[j])$$

where $x[j] = \Sigma_{i=0}^{j} x_i 2^i$ and similarly for $y[j]$ and $p[j]$.

- Both operands used in serial form; the recurrence is

$$\begin{aligned} w[j+1] &= 2^{-(j+2)}(x[j+1] \times y[j+1] - p[j+1]) \\ &= 2^{-(j+2)}((x[j] + x_{j+1}2^{j+1})(y[j] + y_{j+1}2^{j+1}) - (p[j] + p_{j+1}2^{j+1})) \\ &= 2^{-1}(w[j] + y[j+1]x_{j+1} + x[j]y_{j+1} - p_{j+1}) \end{aligned}$$

- This can be expressed as

$$v[j] = w[j] + y[j+1]x_{j+1} + x[j]y_{j+1}$$

and

$$\begin{aligned} w[j+1] &= \lfloor 2^{-1}v[j] \rfloor \\ p_{j+1} &= v[j] mod\ 2 \end{aligned}$$

Cycle	Position 8	7	6	5	4	3	2	1	0
0									y_0x_0
1								x_0y_1	
							y_1x_1	y_0x_1	
2						x_1y_2	x_0y_2		
					y_2x_2	y_1x_2	y_0x_2		
3				x_2y_3	x_1y_3	x_0y_3			
			y_3x_3	y_2x_3	y_1x_3	y_0x_3			
4		x_3y_4	x_2y_4	x_1y_4	x_0y_4				
	y_4x_4	y_3x_4	y_2x_4	y_1x_4	y_0x_4				

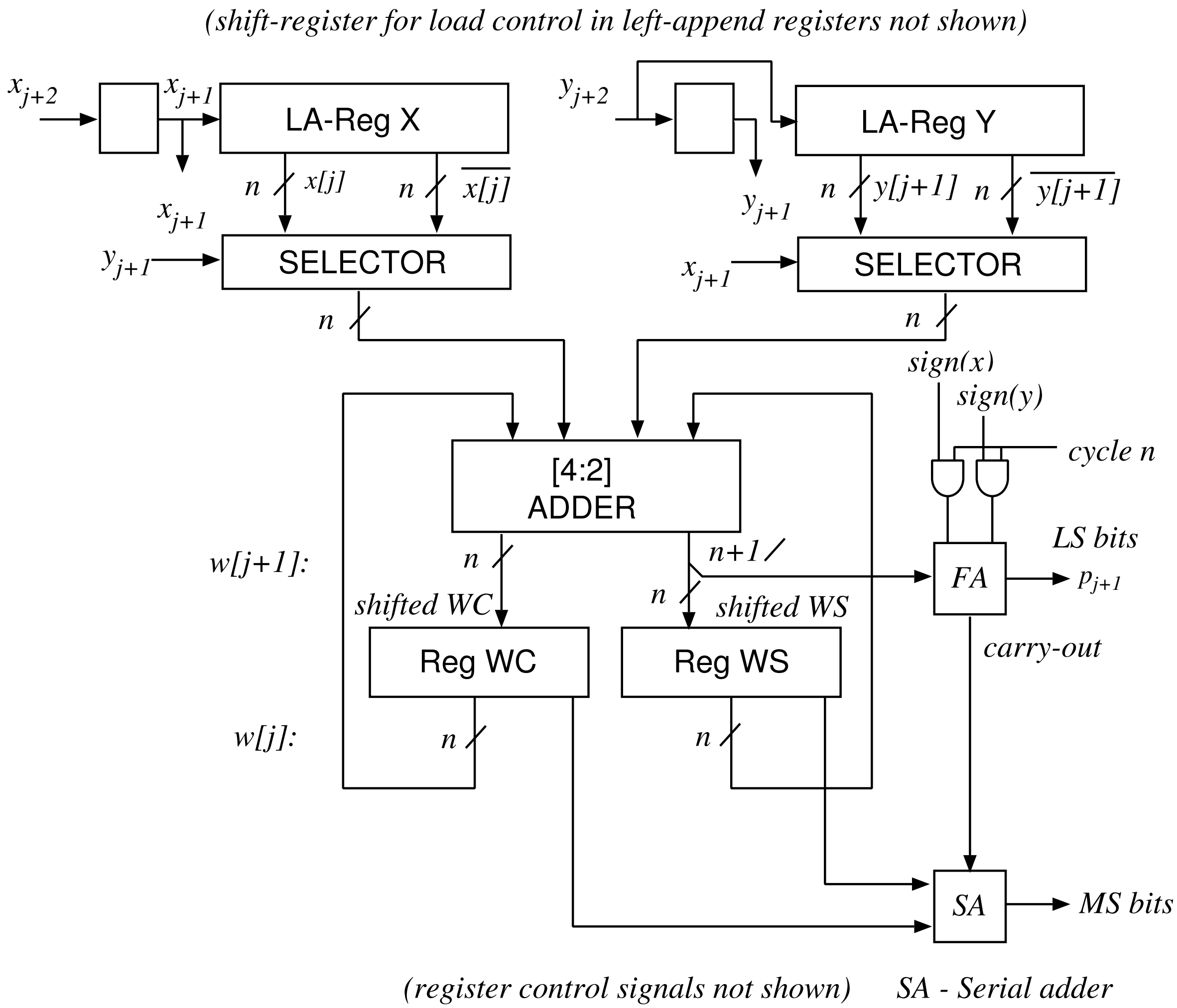


Figure 9.6: Serial-serial 2's complement radix-2 multiplier.

cont.

- The total execution time

$$T_{SSMULT} = 2nt_{cyc}$$

- The delay of the critical path in a cycle

$$t_{cyc} = t_{SEL} + t_{4-2CSA} + t_{FF}$$

- Cost: one n-bit [4:2] adder, 5 n-bit registers, and gates to form multiples

SERIAL-PARALLEL MULTIPLIER

- One of the operands is a constant,

One possibility: perform operation in $3n$ cycles

Phase 1: Serial input and conversion of one operand to parallel form;

Phase 2: Serial-parallel processing and output of the LS half of the product.

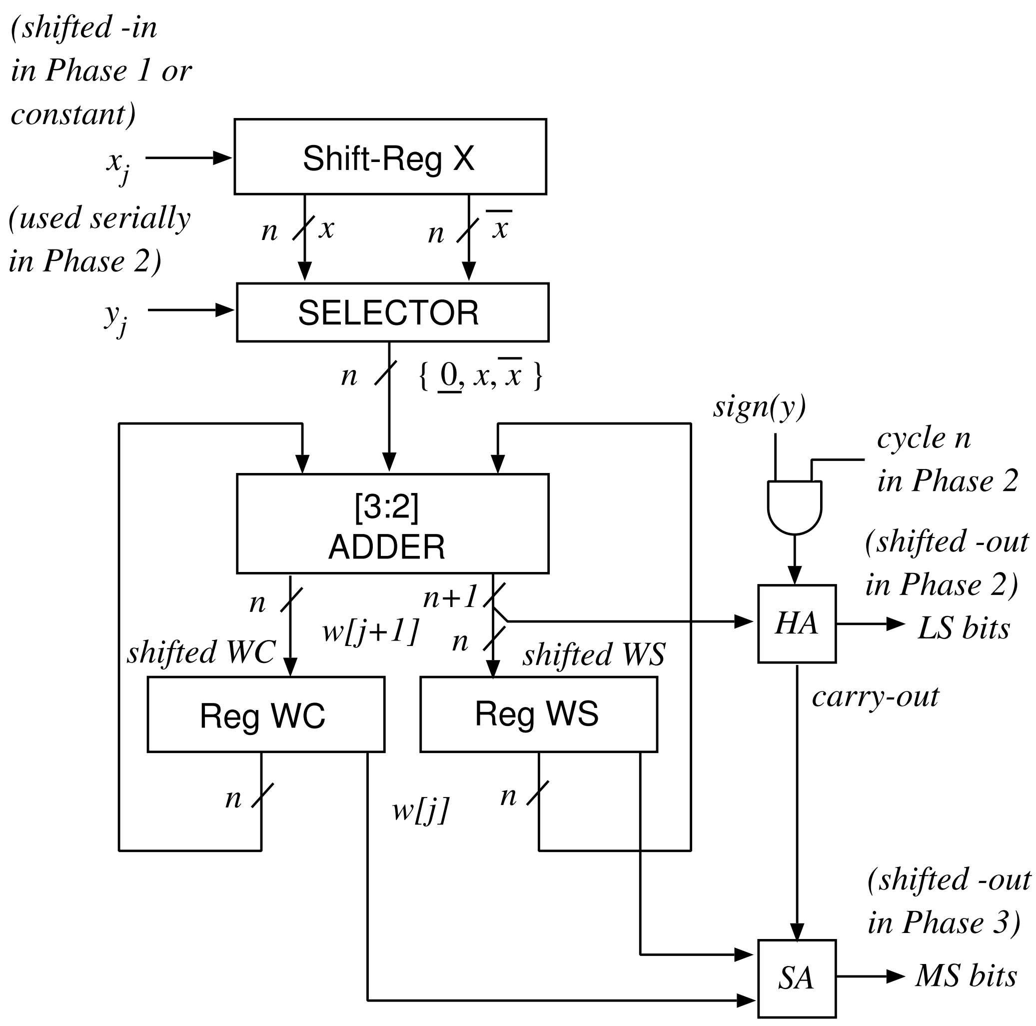
Phase 3: Serial output of the MS half of the product.

- The critical path in a cycle

$$t_{cyc} = t_{SEL} + t_{CSA} + t_{FF}$$

- The delay of the LSDF-SP multiplier

$$T_{SPrnd} = 3n \times t_{cyc}$$



(register control signals not shown)

SA - Serial adder

Phase 1: shift-in operand X (n cycles)
Phase 2: serial-parallel carry-save multiplication (n cycles)
shifted sum and carry bit-vectors loaded bit-parallel
Phase 3: MS bits obtained using bit-serial adder SA operating
on bits shifted out of WC and WS shift-registers (n cycles)

Figure 9.7: 3-phase serial multiplier.

MSDF: ONLINE ARITHMETIC

- Online arithmetic algorithms operate in a digit-serial MSDF mode
- To compute the first digit of the result, $\delta + 1$ digits of the input operands needed
- Thereafter, for each new digit of the operands, an extra digit of the result obtained
- The *online* delay δ typically a small integer, e.g., 1 to 4.

Cycle	-2	-1	0	1	2	$\cdots$
Input	x_1	x_2	x_3	x_4	x_5	$\cdots$
Compute			z_1	z_2	z_3	$\cdots$
Output				z_1	z_2	$\cdots$

$$\delta = 2$$

Figure 9.8: Timing in online arithmetic.

REDUNDANT REPRESENTATION

- The left-to-right mode of computation requires redundancy
- Both symmetric $\{-a, \ldots, a\}$ and asymmetric $\{b, \ldots, c\}$) digit-sets used in online arithmetic
- Over-redundant digit-sets also useful
- Examples of radix-4 redundant digit sets:
  {-1,0,1,2,3} (asymmetric, minimally-redundant),
  {-2,-1,0,1,2} (symmetric, minimally-redundant), and
  {-5,-4,-3,-2,-1,0,1,2,3,4,5} (symmetric, over-redundant)
- Heterogeneous representations to optimize the implementation
- Conversion to conventional representation: use on-the-fly conversion

ADDITION/SUBTRACTION

- The online addition/subtraction algorithm: the serialization of a redundant addition (carry-save or signed-digit)
- Radix $r > 2$

$$(t_{j+1}, w_{j+2}) = \begin{cases} (0, x_{j+2} + y_{j+2}) & \text{if } |x_{j+2} + y_{j+2}| \leq a - 1 \\ (1, x_{j+2} + y_{j+2} - r) & \text{if } x_{j+2} + y_{j+2} \geq a \\ (-1, x_{j+2} + y_{j+2} + r) & \text{if } x_{j+2} + y_{j+2} \leq -a \end{cases}$$

and

- $$z_{j+1} = w_{j+1} + t_{j+1}$$

where $x_j,\ y_j,\ z_j \in \{-a, \ldots, a\}$.

EXAMPLE OF ONLINE ADDITION ($r = 4$, $a = 3$)

- Operands $x = (.12\bar{3}30\bar{1})$ $y = (.2\bar{1}\bar{3}322)$
- The result $z = (1.\bar{1}0\bar{1}221)$.

j	x_{j+2}	y_{j+2}	t_{j+1}	w_{j+2}	w_{j+1}	z_{j+1}	z_j
-1	1	2	1	-1	0*	1	0*
0	2	-1	0	1	-1	-1	1
1	-3	-3	-1	-2	1	0	-1
2	3	3	1	2	-2	-1	0
3	0	2	0	2	2	2	-1
4	-1	2	0	1	2	2	2
5	0	0	0	0	1	1	2
6	0	0	0	0	0	0	1

* latches initialized to 0.

RADIX $r > 2$ ONLINE ADDER

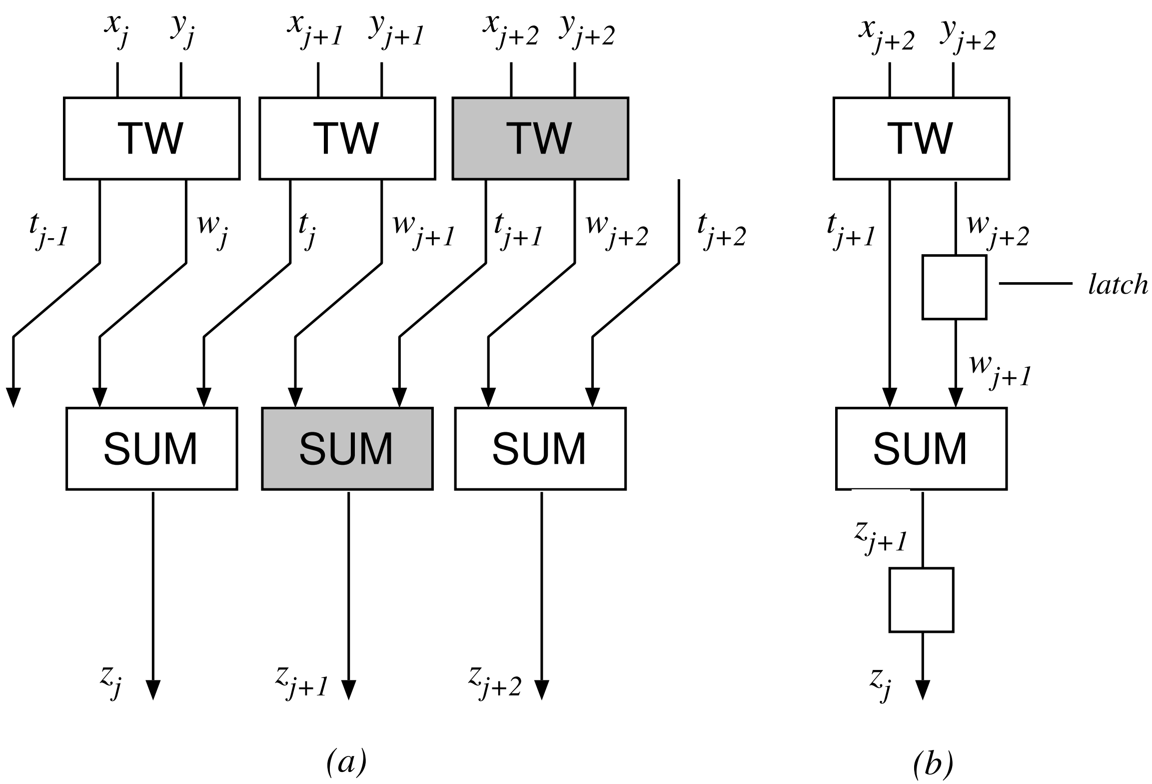


Figure 9.9: (a) A segment of radix-$r > 2$ signed-digit parallel adder. (b) Radix-$r > 2$ online adder. All latches cleared at start.

RADIX-2 ONLINE ADDER

- Digit-parallel radix-2 signed-digit adder converted into a radix-2 online adder with online delay $\delta = 2$

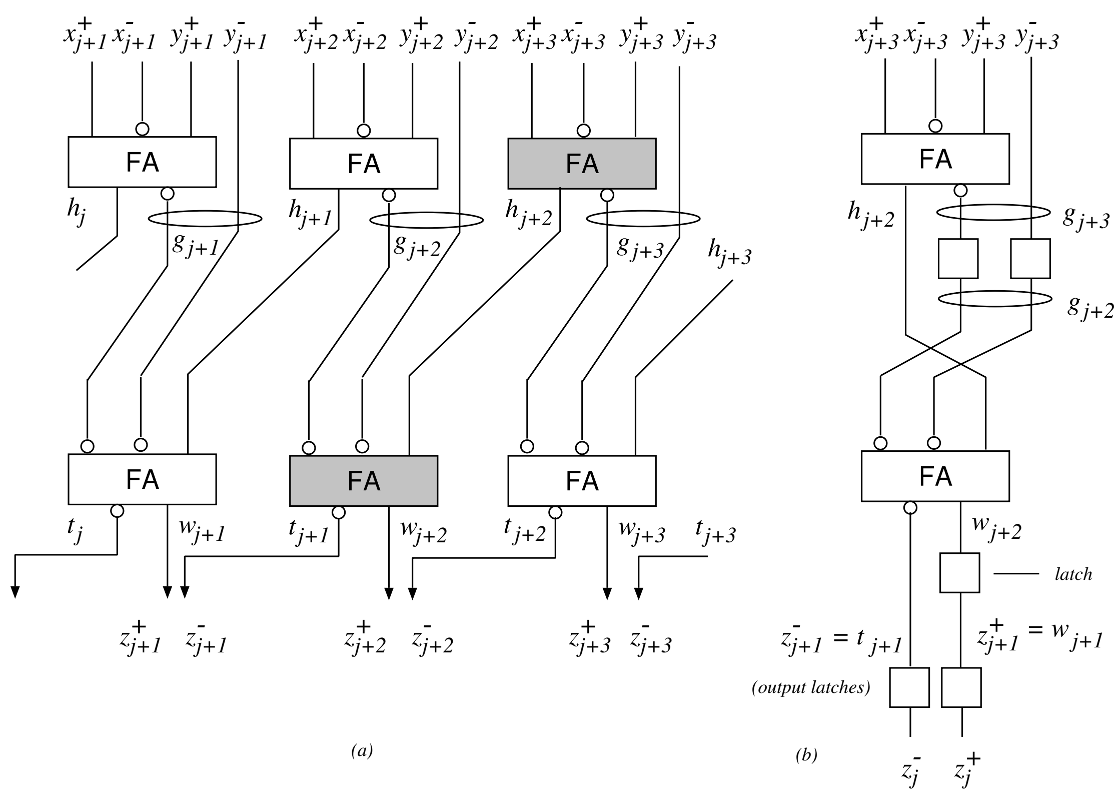


Figure 9.10: (a) A segment of radix-2 signed-digit parallel adder. (b) Online adder.

cont.

- The cycle time is $t_{cyc} = 2t_{FA} + t_{FF}$
- The operation time $T_{OLADD-2} = (2 + n + 1)t_{cyc}$
- The cost 2 FAs and 5 FFs.
- To reduce the cycle time, pipeline the two stages:
  reduces the cycle time by one t_{FA}; increases online delay to $\delta = 3$

EXAMPLE OF RADIX-2 ONLINE ADDITION

$$x = (.010\bar{1}110\bar{1})$$
$$y = (.10\bar{1}01\bar{1}\bar{1}0)$$
$$z = (1.\bar{1}0100\bar{1}01)$$

j	x_{j+3}	y_{j+3}	$x^+_{j+3}x^-_{j+3}$	$y^+_{j+3}y^-_{j+3}$	h_{j+2}	g_{j+3}	g_{j+2}	$t_{j+1}w_{j+2}$	$z^+_{j+1}z^-_{j+1}$	z_j
-2	0	1	00	10	1	10	00*	0 1	- -	-
-1	1	0	10	00	1	10	10	0 0	10	-
0	0	-1	00	01	0	01	10	1 1	01	1
1	-1	0	01	00	0	10	01	1 1	11	-1
2	1	1	10	10	1	00	10	0 0	10	0
3	1	-1	10	01	0	11	00	0 1	00	1
4	0	-1	00	01	0	01	11	1 0	11	0
5	-1	0	01	00	0	10	01	1 1	01	0
6	0	0	00	00	0	00	10	1 1	11	-1
7	0	0	00	00	0	00	00	0 0	10	0
8	0	0	00	00	0	00	00	0 0	00	1

* g latches initialized to 00.

METHOD FOR DEVELOPING ONLINE ALGORITHMS

- Part 1: development of the residual recurrence

$$w[j+1] = G(w[j], x[j], x_{j+1+\delta}, y[j], y_{j+1+\delta}, z[j], z_{j+1})$$

for $-\delta \leq j \leq n-1$ where

$$x[j] = \sum_{i=1}^{j+\delta} x_i r^{-i}, \;\; y[j] = \sum_{i=1}^{j+\delta} y_i r^{-i}, \;\; z[j] = \sum_{i=1}^{j} z_i r^{-i}$$

are the online forms of the operands and the result

- Part 2: the result digit selection

$$z_{j+1} = F(w[j], x[j], x_{j+1+\delta}, y[j], y_{j+1+\delta}, z[j])$$

Part 1: RESIDUAL AND ITS RECURRENCE

Step 1. Describe the online operation by the error bound after j digits

$$|f(x[j], y[j]) - z[j]| < r^{-j}$$

Step 2 Transform expression to use only

- multiplication by r (shift),
- addition/subtraction,
- multiplication by a single digit

$$\underline{B} < G(f(x[j],\ y[j]) - z[j]) < \overline{B}$$

where G is the required transformation and $\underline{B}$ and $\overline{B}$ are the transformed bounds

Example: division error expression $|x[j]/y[j] - z[j]| < r^{-j}$ transformed into

$$|x[j] - z[j] \cdot y[j]| < |r^{-j} y[j]|$$

cont.

Step 3 Define a scaled residual

$$w[j] = r^j(G(f(x[j], y[j]) - z[j]))$$

with the bound

$$\underline{\omega} = r^j \underline{B} < w[j] < r^j \overline{B} = \overline{\omega}$$

and initial condition $w[-\delta] = 0$. $\underline{\omega}$ and $\overline{\omega}$ are the actual bounds determined in Step 6

Step 4 Determine a recurrence on $w[j]$

$$w[j+1] = rw[j] + r^{j+1}(G(f(x[j+1], y[j+1]) - z[j+1]) - G(f(x[j], y[j]) - z[j]))$$

Step 5 Decompose recurrence so that H_1 is independent of z_{j+1}

$$w[j+1] = rw[j] + H_1 + H_2(z_{j+1}) = v[j] + H_2(z_{j+1})$$

cont.

Step 6 Determine the bounds of $w[j+1]$ in terms of H_1 and H_2

$$\overline{\omega} = r\overline{\omega} + max(H1) + H2(a)$$

resulting in

$$\overline{\omega} = -\frac{\max(H_1) + H_2(a)}{r-1}$$

Similarly,

$$\underline{\omega} = -\frac{\min(H_1) + H_2(-a)}{r-1}$$

Part 2a: SELECTION FUNCTION WITH SELECTION CONSTANTS

$$z_{j+1} = k \text{ if } m_k \leq \widehat{v}[j] < m_{k+1}$$

where $\widehat{v}[j]$ is an estimate of $v[j]$ obtained by truncating the redundant representation of $v[j]$ to t fractional bits.

Selection constants need to satisfy

$$\max(\widehat{L}_k) \leq m_k \leq \min(\widehat{U}_{k-1})$$

where $[\widehat{L}_k,\ \widehat{U}_k]$ is the selection interval of the estimate $\widehat{v}[j]$

Step 7 Determine $[\widehat{L}_k,\ \widehat{U}_k]$

First, determine $[L_k,\ U_k]$ for $v[j]$

$$\overline{\omega} = U_k + H_2(k) \quad \underline{\omega} = L_k + H_2(k)$$

Substituting $\overline{\omega}$ and $\underline{\omega}$ the selection intervals for $v[j]$ is,

$$\begin{aligned} U_k &= -\tfrac{max(H_1)+H_2(a)}{r-1} - H_2(k) \\ L_k &= -\tfrac{min(H_1)+H_2(-a)}{r-1} - H_2(k) \end{aligned}$$

cont.

Now restrict the intervals because of the use of the estimate $\widehat{v}[j]$

$$e_{min} \leq v[j] - \widehat{v}[j] \leq e_{max}$$

producing the error-restricted selection interval $[L_k^*,\ U_k^*]$ with

$$U_k^* = U_k - e_{max} \quad L_k^* = L_k + |e_{min}|$$

The errors are

- For carry-save representation $e_{max} = 2^{-t+1} - ulp$ and $e_{min} = 0$.
- For signed-digit representation $e_{max} = 2^{-t} - ulp$ and $e_{min} = -(2^{-t} - ulp)$.

$$\begin{aligned} \widehat{U}_{k-1} &= \lfloor U_{k-1}^* + 2^{-t} \rfloor_t \\ \widehat{L}_k &= \lceil L_k^* \rceil_t \end{aligned}$$

where $\lfloor x \rfloor_t$ and $\lceil x \rceil_t$ indicate x values truncated to t fractional bits.

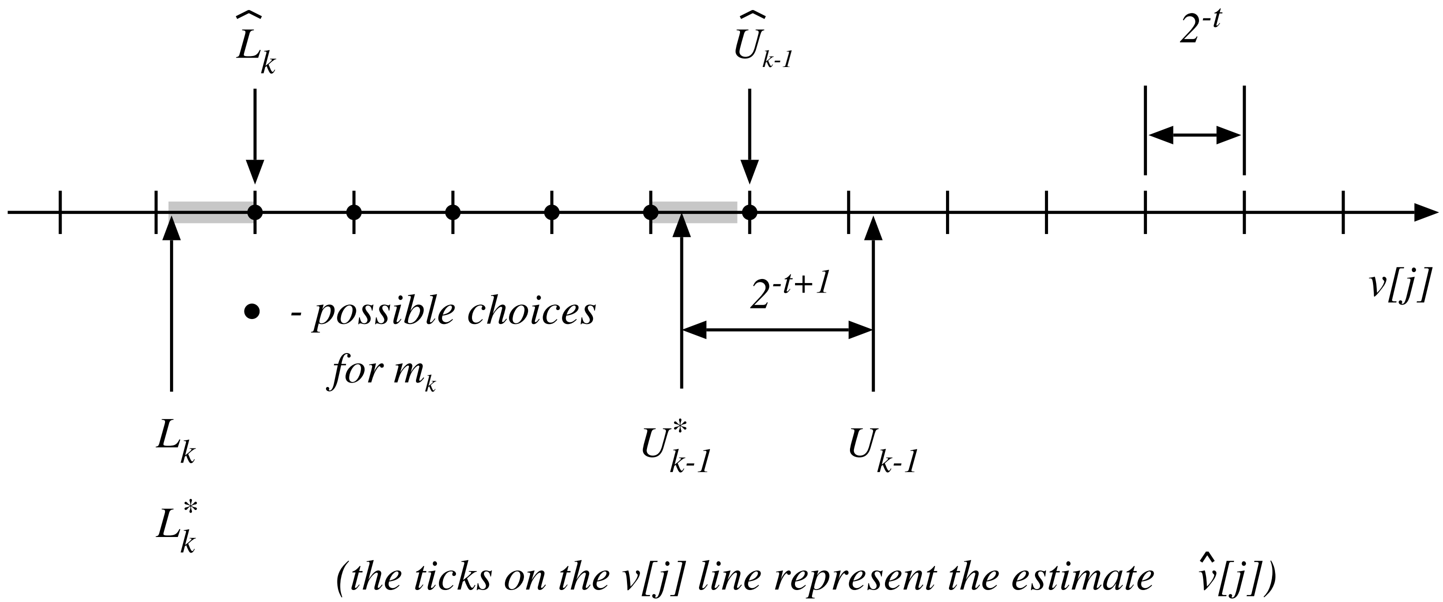


Figure 9.11: The choices of selection constant m_k.

cont.

Step 8 Determine t and δ. To determine m_k, we need

$$\min(\widehat{U}_{k-1}) - \max(\widehat{L}_k) \geq 0$$

This relation between t and δ is used to choose suitable values.

Step 9 Determine the selection constants m_k and the range of $\widehat{v}[j]$ as

$$\lfloor r\underline{\omega} + \min(H_1) - e_{max} \rfloor_t \leq \widehat{v}[j] \leq \lfloor r\overline{\omega} + \ max(H_1) + |e_{min}| \rfloor_t$$

Part 2b: SELECTION BY ROUNDING

- In algorithms using a higher radix ($r > 4$)

$$w[j+1] = rw[j] + H_1 + H_2(z_{j+1}) = v[j] + H_2(z_{j+1})$$

- In the rounding method, the result digit is obtained as

$$z_{j+1} = \lfloor v[j] + \frac{1}{2} \rfloor$$

with $|v[j]| < r - \frac{1}{2}$ to avoid over-redundant output digit.

$$w[j+1] = v[j] + H_2(\lfloor v[j] + \frac{1}{2} \rfloor)$$

- For CS form of t fractional bits, the estimate error

$$e_{max} = 2^{-t+1} - ulp$$

- When $\hat{v}[j] = m_k - 2^{-t}$ it must be possible to choose $z_{j+1} = k - 1$

$$m_k - 2^{-t} + e_{max} = \frac{2k-1}{2} + 2^{-t} \leq \widehat{U}_{k-1}$$

GENERIC FORM OF EXECUTION AND IMPLEMENTATION.

- Execution: $n + \delta$ iterations of the recurrence, each one clock cycle
- Iterations (cycles) labeled from $-\delta$ to $n - 1$
- One digit of each input introduced during cycles $-\delta$ to $n - 1 - \delta$ and digits value 0 thereafter
- Result digits 0 for cycles $-\delta$ to -1 and z_1 is produced in cycle 0
- Result digit z_j is output in cycle j (one extra cycle to output z_n)

cont.

- The actions in cycle j:
  - Input $x_{j+1+\delta}$ and $y_{j+1+\delta}$.
  - Update $x[j+1] = (x[j], x_{j+1+\delta})$ and $y[j+1] = (y[j], y_{j+1+\delta})$ by appending the input digits.
  - Compute $v[j] = rw[j] + H_1$
  - Determine z_{j+1} using the selection function.
  - Update $z[j+1] = (z[j], z_{j+1+\delta})$ by appending the result digits.
  - Compute the next residual $w[j+1] = v[j] + H_2(z_{j+1})$
  - Output result digit z_j

IMPLEMENTATION

- Similar structure of algorithms $\rightarrow$ all implemented with same basic components, such as

  (i) registers to store operands, results, and residual vectors;

  (ii) multiplication of vector by digit;

  (iii) append units to append a new digit to a vector;

  (iv) Two-operand and multioperand redundant adders, such as signed digit adders, [3:2] carry-save adders and their generalization to [4:2] and [5:2] adders;

  (v) converters from redundant representations (i.e., signed digit and carry save) to conventional representations;

  (vi) carry-propagate adders of limited precision (3 to 6 bits) to produce estimates of the residual functions; and

  (vii) digit-selection schemes to obtain output digits.

cont.

- Online algorithm implementation similar to implementation of digit-recurrence algorithms
- Algorithms and implementations developed for most of basic arithmetic operations and for certain composite operations
- Larger set of operations possible than with LSDF approach

DIGIT-SLICE ORGANIZATION

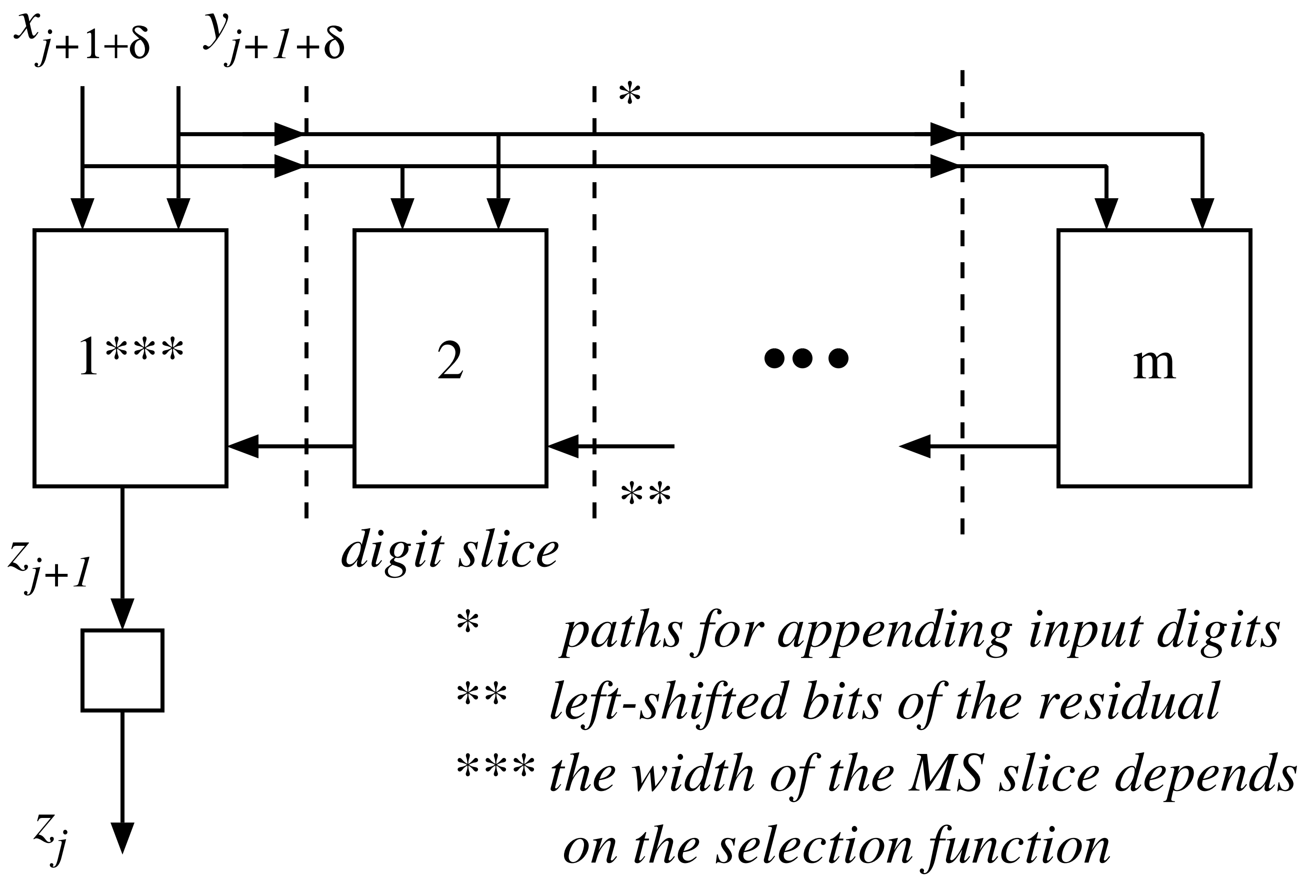


Figure 9.12: A typical digit-slice organization of online arithmetic unit

ONLINE MULTIPLICATION

- Online forms

$$x[j] = \sum_{i=1}^{j+\delta} x_i r^{-i}, \; y[j] = \sum_{i=1}^{j+\delta} y_i r^{-i}, \; p[j] = \sum_{i=1}^{j} p_i r^{-i}$$

- The error bound at cycle j

$$|x[j] \cdot y[j] - p[j]| < r^{-j}$$

- The residual

$$w[j] = r^j (x[j] \cdot y[j] - p[j])$$

with the bound $|w[j]| < \omega$

- The residual recurrence

$$\begin{aligned} w[j+1] &= rw[j] + (x[j]y_{j+1+\delta} + y[j+1]x_{j+1+\delta})r^{-\delta} - p_{j+1} \\ &= v[j] - p_{j+1} \end{aligned}$$

SELECTION FUNCTION

- Decomposition

$$H_1 = (x[j]y_{j+1+\delta} + y[j+1]x_{j+1+\delta})r^{-\delta} \quad H_2 = -p_{j+1}$$

- Bound

$$\overline{\omega} = -\underline{\omega} = \omega = \rho(1-2r^{-\delta})$$

- Selection intervals

$$\begin{aligned} U_k &= \rho(1-2r^{-\delta}) + k \\ L_k &= -\rho(1-2r^{-\delta}) + k \end{aligned}$$

- With carry-save representation for $w[j]$ and $v[j]$, the grid-restricted intervals are

$$\begin{aligned} \widehat{U}_k &= \lfloor \rho(1-2r^{-\delta}) + k - 2^{-t} \rfloor_t \\ \widehat{L}_k &= \lceil -\rho(1-2r^{-\delta}) + k \rceil_t \end{aligned}$$

- The expression to determine t and δ:

$$\lfloor \rho(1-2r^{-\delta}) + k - 1 - 2^{-t} \rfloor_t - \lceil -\rho(1-2r^{-\delta}) + k \rceil_t \geq 0$$

resulting in

$$\lfloor \rho(1-2r^{-\delta}) \rfloor_t \geq 2^{-1}(1+2^{-t})$$

cont.

- Several examples of relations between r, ρ, t, and δ

Radix	ρ	t	δ
2	1	2	3
4	1	2	2
	2/3	3	3
8	2/3	2	3

RADIX-2 ONLINE MULTIPLICATION

- $\delta = 3$ and $t = 2$
- Selection constants m_k's obtained from

$$\widehat{L}_k \leq m_k \leq \widehat{U}_{k-1}$$

where

$$\begin{aligned} \widehat{U}_k &= \lfloor 1 - 2^{-2} + k - 2^{-2} \rfloor_2 = k + 2^{-1} \\ \widehat{L}_k &= \lceil -1 + 2^{-2} + k \rceil_2 = k - 3 \times 2^{-2} \end{aligned}$$

- Since $\widehat{U}_{k-1} = k - 2^{-1}$ and $\widehat{L}_k = k - 3 \times 2^{-2}$, $m_k = k - 2^{-1}$ is acceptable. The selection constants are

$$m_0 = -2^{-1}, \quad m_1 = 2^{-1}$$

- Range of $\widehat{v}[j]$ is

$$-2 \leq \widehat{v}[j] \leq 7/4$$

- The selection function $SELM(\widehat{v}[j])$ is

$$p_{j+1} = SELM(\widehat{v}[j]) = \begin{cases} 1 & \textbf{if} \;\; 1/2 \leq \widehat{v}[j] \leq 7/4 \\ 0 & \textbf{if} \;\; -1/2 \leq \widehat{v}[j] \leq 1/4 \\ -1 & \textbf{if} \;\; -2 \leq \widehat{v}[j] \leq -3/4 \end{cases}$$

IMPLEMENTATION OF SELECTION FUNCTION

- Estimate $\hat{v}$ represented by (v_{-1}, v_0, v_1, v_2)
- Product digit $p_{j+1} = (pp, pn)$ with the code

p_{j+1}	pp	pn
1	1	0
0	0	0
-1	0	1

- Switching expressions:

$$pp = v'_{-1}(v_0 + v_1)$$
$$pn = v_1(v'_0 + v'_1)$$

$\hat{v}$	$v_{-1}v_0v_1v_2$	p_{j+1}
7/4	01.11	1
6/4	01.10	1
5/4	01.01	1
1	01.00	1
3/4	00.11	1
1/2	00.10	1
1/4	00.01	0
0	00.00	0
-1/4	11.11	0
-1/2	11.10	0
-3/4	11.01	-1
-1	11.00	-1
-5/4	10.11	-1
-6/4	10.10	-1
-7/4	10.01	-1
-2	10.00	-1

1. $[Initialize]$

   $x[-3] = y[-3] = w[-3] = 0$

   for $j = -3, -2, -1$

   $x[j+1] \leftarrow CA(x[j], x_{j+4});\ \ y[j+1] \leftarrow CA(y[j], y_{j+4})$

   $v[j] = 2w[j] + (x[j]y_{j+4} + y[j+1]x_{j+4})2^{-3}$

   $w[j+1] \leftarrow v[j]$

   end for

2. $[Recurrence]$

   for $j = 0 \ldots n-1$

   $x[j+1] \leftarrow CA(x[j], x_{j+4});\ \ y[j+1] \leftarrow CA(y[j], y_{j+4})$

   $v[j] = 2w[j] + (x[j]y_{j+4} + y[j+1]x_{j+4})2^{-3}$

   $p_{j+1} = SELM(\widehat{v[j]});$

   $w[j+1] \leftarrow v[j] - p_{j+1}$

   $P_{out} \leftarrow p_{j+1}$

   end for

Figure 9.13: Radix-2 online multiplication algorithm.

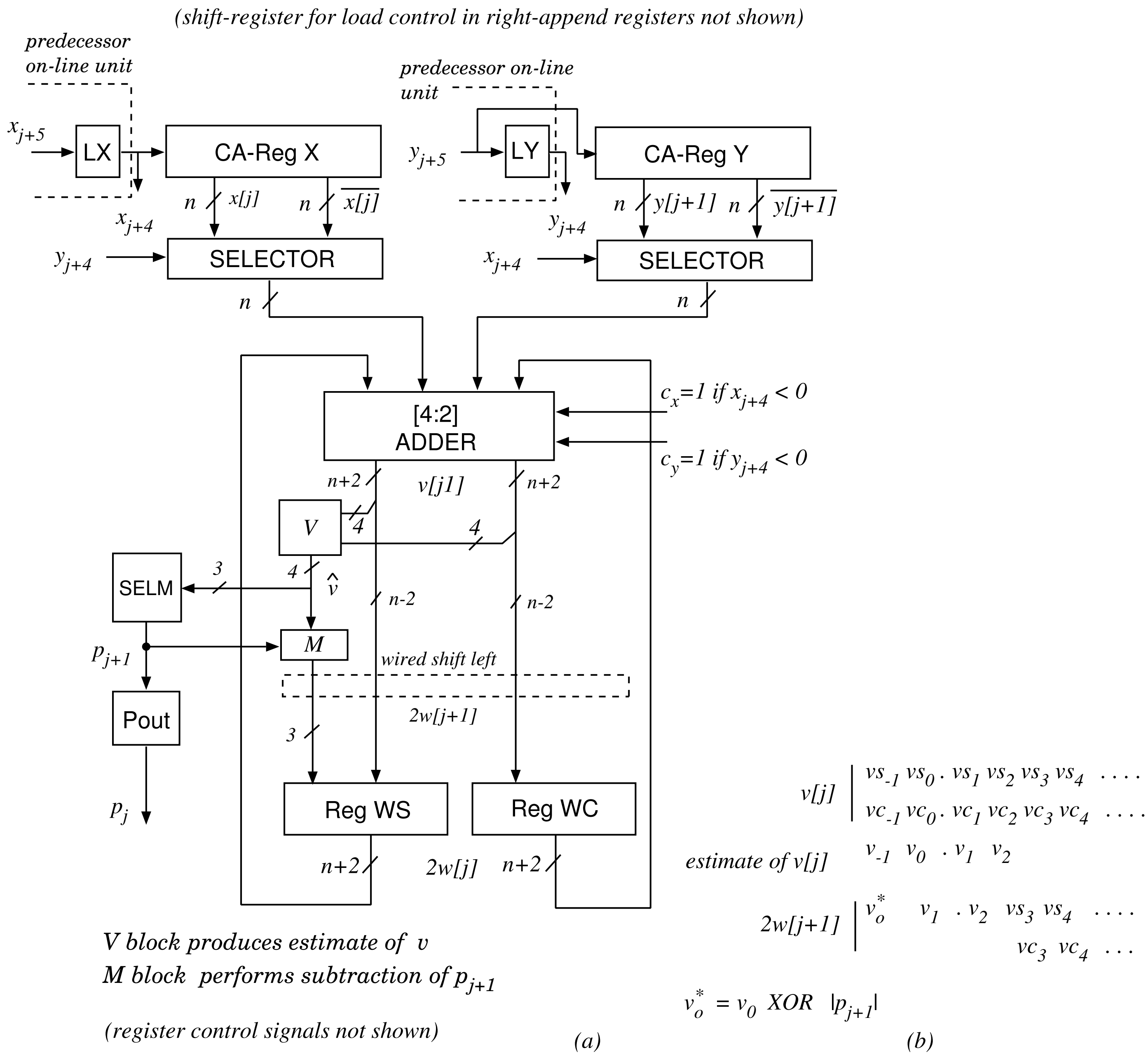


Figure 9.14: (a) Implementation of radix-2 online multiplier. (b) Calculation of $2w[j+1]$.

EXAMPLE OF RADIX-2 ONLINE MULTIPLICATION

Operands:

$$x = (.110\bar{1}10\bar{1}1)$$
$$y = (.101\bar{1}\bar{1}110)$$

j	x_{j+4}	y_{j+4}	$x[j+1]$	$y[j+1]$	$v[j]$	p_{j+1}	$w[j+1]$
-3	1	1	.1	.1	00.0001	0	00.0001
-2	1	0	.11	.10	00.00110	0	00.00110
-1	0	1	.110	.101	00.011110	0	00.011110
0	-1	-1	.1011	.1001	00.1100011	1	11.1100011
1	1	-1	.10111	.10001	11.10000111	0	11.10000111
2	0	1	.101110	.100011	11.001001010	-1	00.001001010
3	-1	1	.1011011	.1000111	00.0100111101	0	00.0100111101
4	1	0	.10110111	.10001110	00.10110000010	1	11.10110000010
5	0	0	.10110111	.10001110	11.0110000010	-1	00.0110000010
6	0	0	.10110111	.10001110	00.110000010	1	11.110000010
7	0	0	.10110111	.10001110	11.10000010	0	11.10000010

cont.

- Computed product: $p = (.10\bar{1}01\bar{1}10)$
- The exact double precision product $p^* = (.0110010110000010)$
- The absolute error wrt to the exact product truncated to 8 bits:

$$|p - p^*_{tr}| = 2^{-8}$$

- Note: $p[8] + w[8]2^{-8} = p^*$

ONLINE DIVISION

- Online forms

$$x[j] = \sum_{i=1}^{j+\delta} x_i r^{-i}, \; y[j] = \sum_{i=1}^{j+\delta} y_i r^{-i}, \; q[j] = \sum_{i=1}^{j} q_i r^{-i}$$

- Error bound at cycle j

$$|x[j] - q[j]d[j]| < d[j]r^{-j}$$

- Residual

$$w[j] = r^j(x[j] - q[j]d[j]) \quad \|w[j]\| < \omega \leq d[j]$$

- Residual recurrence

$$\begin{aligned} w[j+1] &= rw[j] + x_{j+1+\delta}r^{-\delta} - q[j]d_{j+1+\delta}r^{-\delta} - d[j+1]q_{j+1} \\ &= v[j] - d[j+1]q_{j+1} \end{aligned}$$

RADIX-2 ONLINE DIVISION

- $\delta = 4$ and $t = 3$
- Selection intervals and selection constants

$$\begin{aligned}\min \widehat{U}_0 &= \widehat{U}_0[d[j+1] = 1/2] = 2^{-1} - 2^{-3} + 0 - 2^{-3} = 2^{-2} \\ \max \widehat{L}_1 &= \widehat{L}_1[d[j+1] = 1] = -1 + 2^{-3} + 1 = 2^{-3}\end{aligned}$$

resulting in $m_1 = 2^{-2}$

$$\begin{aligned}\min \widehat{U}_{-1} &= \widehat{U}_{-1}[d[j+1] = 1] = 1 - 2^{-3} - 1 - 2^{-3} = -2^{-2} \\ \max \widehat{L}_0 &= \widehat{L}_0[d[j+1] = 1/2] = -2^{-1} + 2^{-3} = -3 \times 2^{-3}\end{aligned}$$

so that $m_0 = -2^{-2}$.

$$q_{j+1} = SELD(\widehat{v}[j]) = \begin{cases} 1 & \textbf{if} \quad 1/4 \le \widehat{v}[j] \le 15/8 \\ 0 & \textbf{if} \quad -1/4 \le \widehat{v}[j] \le 1/8 \\ -1 & \textbf{if} \quad -2 \le \widehat{v}[j] \le -1/2 \end{cases}$$

RADIX-2 ONLINE DIVISION: ALGORITHM

1. $[Initialize]$

   $x[-4] = d[-4] = w[-4] = q[0] = 0$

   for $j = -4, \ldots, -1$

   $d[j+1] \leftarrow CA(d[j], d_{j+5})$

   $v[j] = 2w[j] + x_{j+5}2^{-4}$

   $w[j+1] \leftarrow v[j]$

   end for

2. $[Recurrence]$

   for $j = 0 \ldots n-1$

   $d[j+1] \leftarrow CA(d[j], d_{j+5})$

   $v[j] = 2w[j] + x_{j+5}2^{-4} - q[j]d_{j+5}2^{-4}$

   $q_{j+1} = SELD(\widehat{v}[j]);$

   $w[j+1] \leftarrow v[j] - q_{j+1}d[j+1]$

   $q[j+1] \leftarrow CA(q[j], q_{j+1})$

   $Q_{out} \leftarrow q_{j+1}$

   end for

(shift-register for load control in right-append registers not shown)

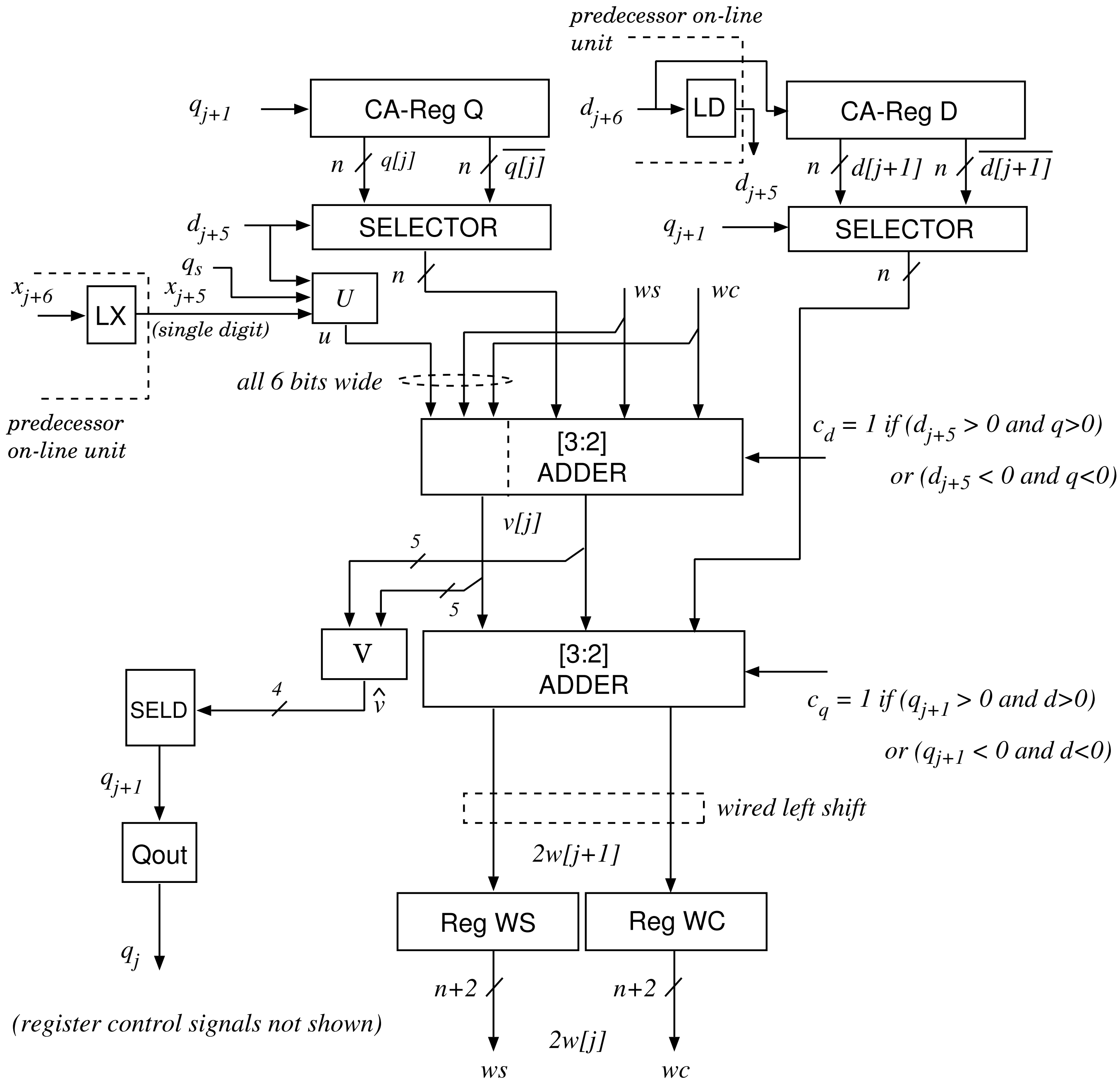


Figure 9.16: Block diagram of radix-2 online divider.

REDUCTION OF DIGIT-SLICES

- Selection valid if (t fractional bits)

$$p - 2h + \delta \geq t$$

- $p + h = n + \delta$

$$p = \left\lceil \frac{2n + \delta + t}{3} \right\rceil$$

- Total number of bit-slices: $ib + p$, ib - no. integer bits
- For example, the number of bit-slices for 32-bit radix-2 online multiplication is

$$2 + \left\lceil \frac{2 \times 32 + 3 + 2}{3} \right\rceil = 2 + 23 = 25$$

compared to 34 in implementation without slice reduction.

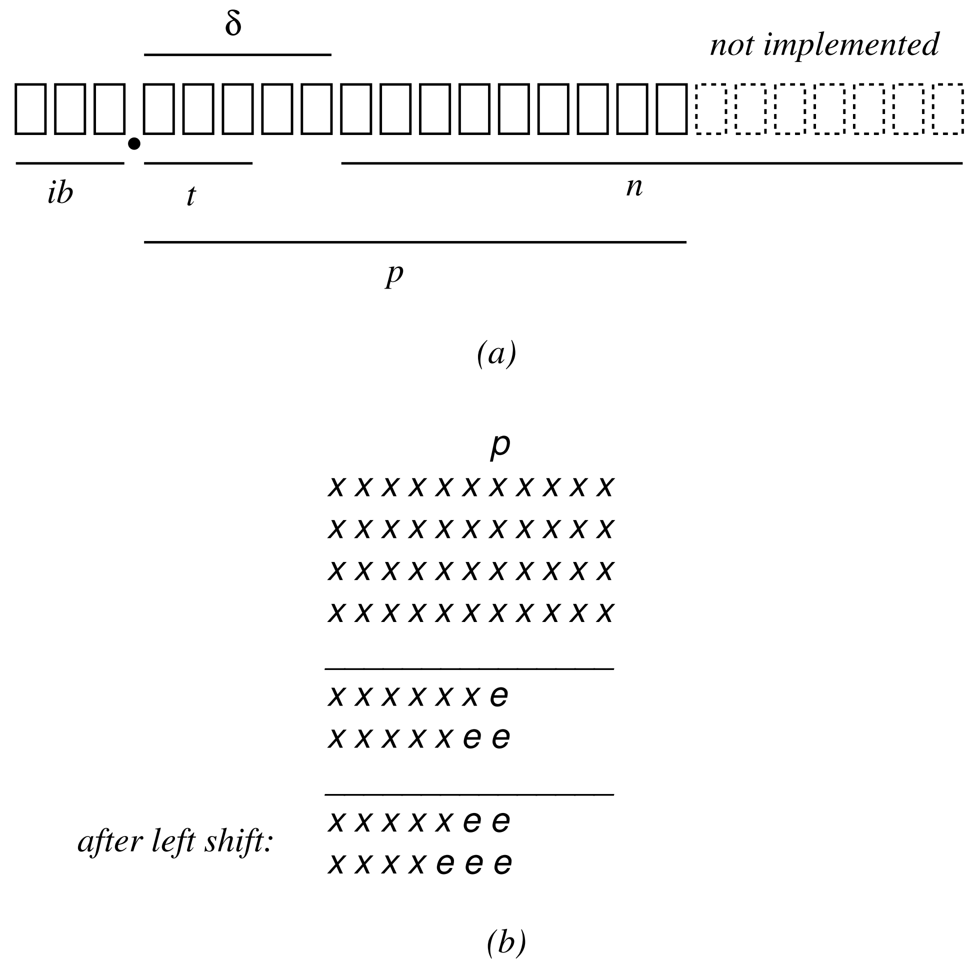


Figure 9.17: Reduction of bit-slices in implementation.

MULTI-OPERATION AND COMPOSITE ONLINE ALGORITHMS

- To reduce the overall online delay of a group of operations
  - combine several operations into a single *multi-operation online algorithm*
- Example: $x^2 + y^2 + z^2$
- Inputs in [1/2,1), output in [1/4, 3)
- Online delay $\delta_{ss} = 0$ when the output digit is over-redundant.
- Online delay (3+2+2=7) of the corresponding network

ALGORITHM FOR SUM OF SQUARES

1. $[Initialize]$

   $w[0] = x[0] = y[0] = z[0] = 0$
2. $[Recurrence]$

   for $j = 0 \ldots n-1$

   $v[j] = 2w[j] + (2x[j] + x_j 2^{-j})x_j + (2y[j] + y_j 2^{-j})y_j + (2z[j] + z_j 2^{-j})z_j$

   $w[j+1] \leftarrow csfract(v[j])$

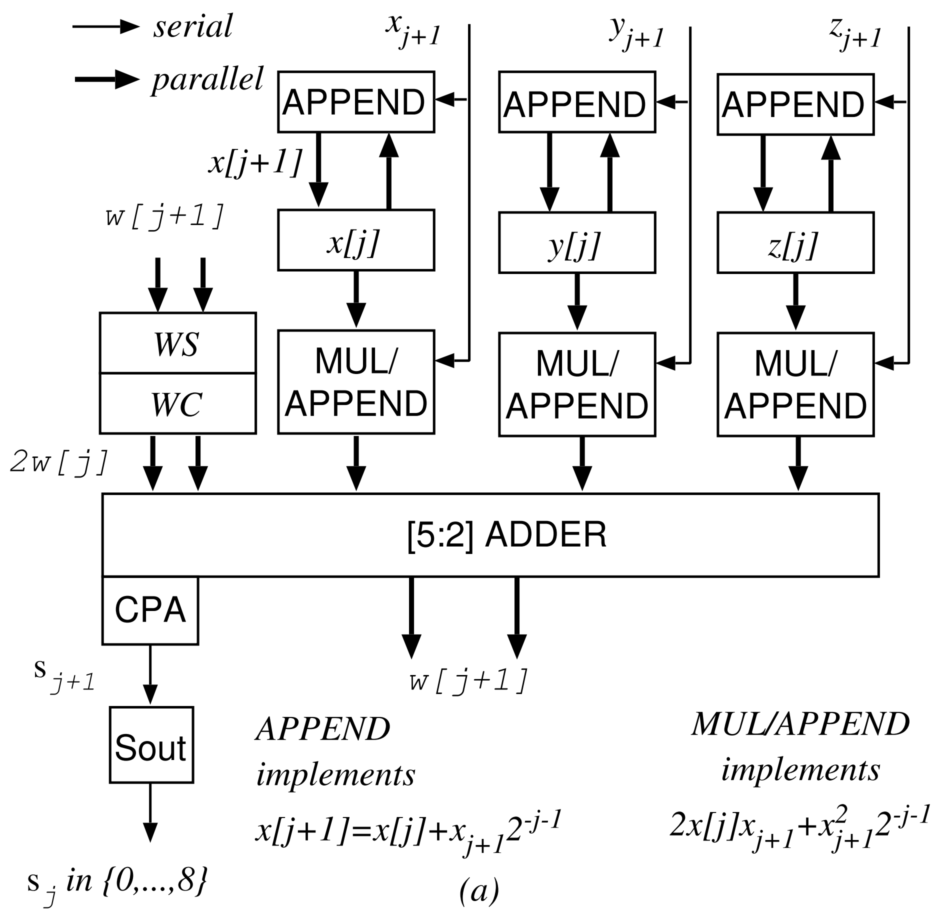
   $s_{j+1} \leftarrow csint(v[j])$

   $x[j+1] \leftarrow (x[j], x_{j+1});\;\; y[j+1] \leftarrow (y[j], y_{j+1});\;\; z[j+1] \leftarrow (z[j], z_{j+1})$

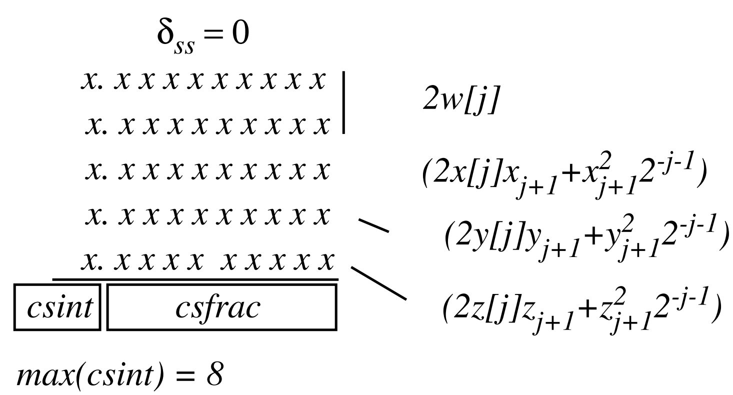
   $S_{out} \leftarrow s_{j+1}$

   end for

Figure 9.18: Radix-2 online sum of squares algorithm.



(a)



Note: the fractional portion of the 5-2 CSA produces at most three carries

COMPOSITE ALGORITHM

- $d = \sqrt{(x^2 + y^2 + z^2)}$
- Overall online delay of 5
- A network of standard online modules: online delay of 11

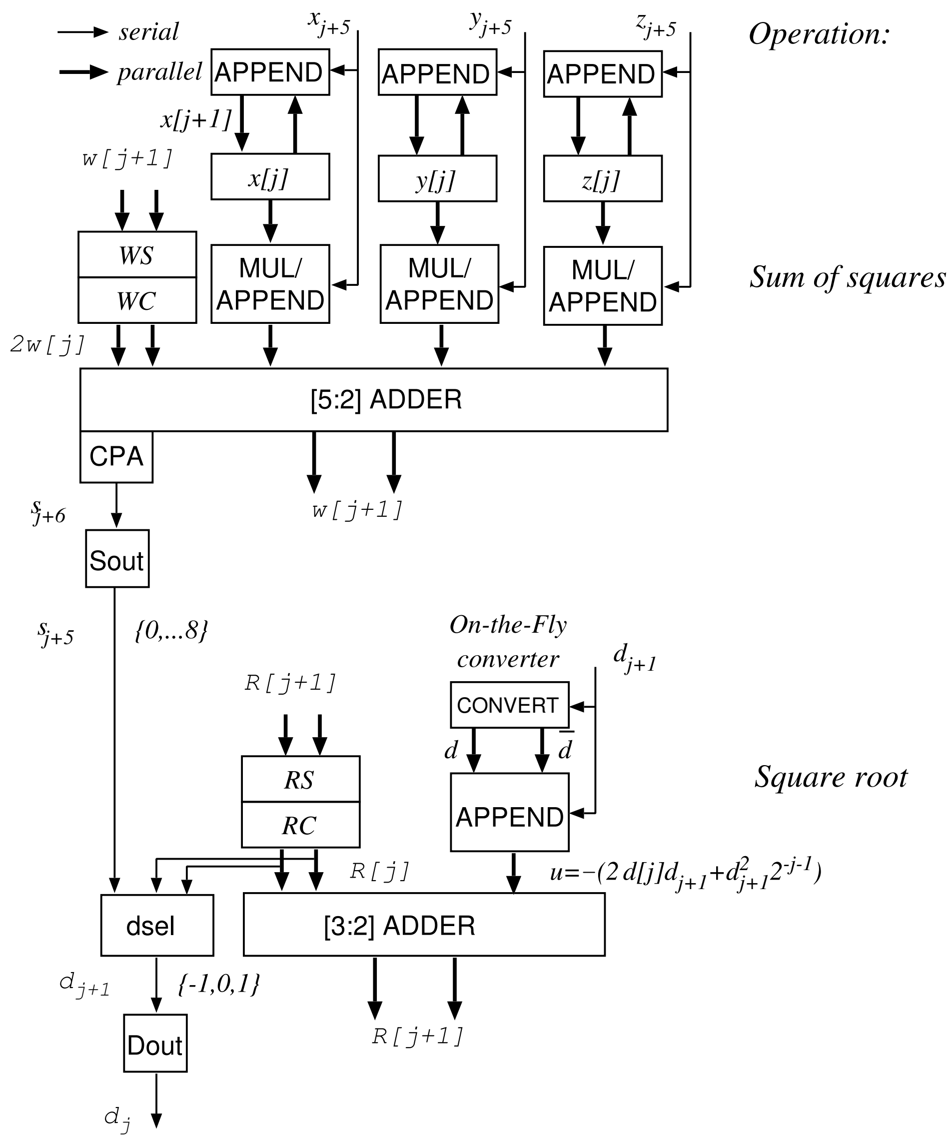


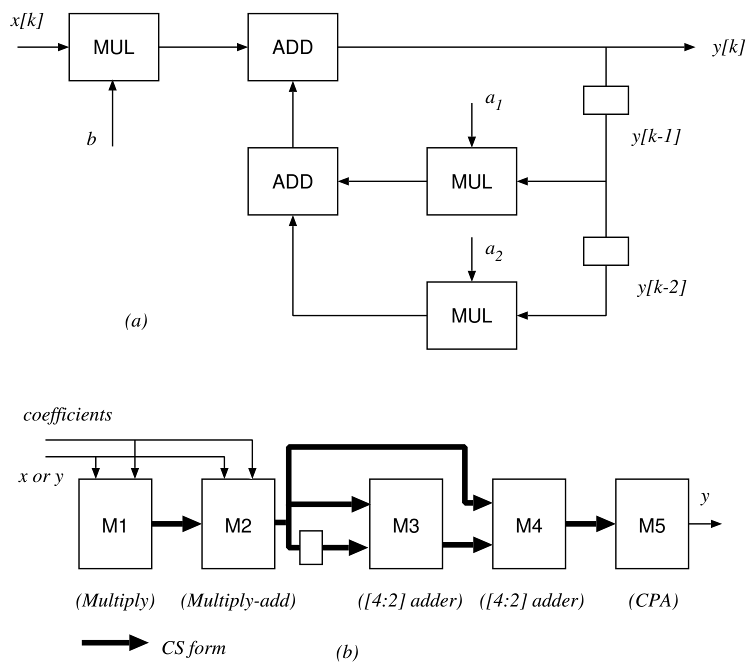
Figure 9.20: Composite scheme for computing $d = \sqrt{(x^2 + y^2 + z^2)}$.

ONLINE IMPLEMENTATION OF RECURSIVE FILTER

- IIR filter

$$y(k) = a_1 y(k-1) + a_2 y(k-2) + bx(k)$$

- **Conventional parallel arithmetic**
  - time to obtain $y[k]$: $T_{CONV} = 6t_{module}$.
  - $t_{module} \approx \ 6t_{FA}$
  - rate of filter computation: $R_{CONV} \approx 1/(4 \times 6t_{FA})$
- LSDF serial arithmetic
  - time to obtain $y[k]$: $T_{LSDF} = nt_{FA}$.
  - rate of filter computation: $R_{LSDF} \approx 1/(n \times t_{FA})$
- **Online arithmetic**
  - Multioperation modules of type $vu + w$, online delay of 4
  - cycle time $t_M \approx 3t_{FA}$
  - Throughput independent of working precision but not the number of online units
  - Rate: $R_{OL} = 1/(\Delta_{iter} \times t_M) \approx 1/(12t_{FA})$



CYCLE: / Module:	k	$k+1$	$k+2$	$k+3$	$k+4$	$k+5$
M1	$bx[k]_R$	$a_2y[k-2]_R$	$a_1y[k-1]_R$		$bx[k+1]_R$	
M2		$(bx[k]_R)$ $+bx[k]_L$	$(a_2y[k-2]_R)+$ $a_2y[k-2]_L$	$(a_1y[k-1]_R)+$ $a_1y[k-1]_L$		
M3				$(bx[k]) +$ $(a_2y[k-2])$		
M4					$(bx[k] +$ $a_2y[k-2])+$ $(a_1y[k-1])$	
M5						$y[k]$

(c)

Figure 9.21: Conventional implementation of second-order IIR filter: (a) Filter. (b) 5-stage pipeline. (c) Timing diagram.

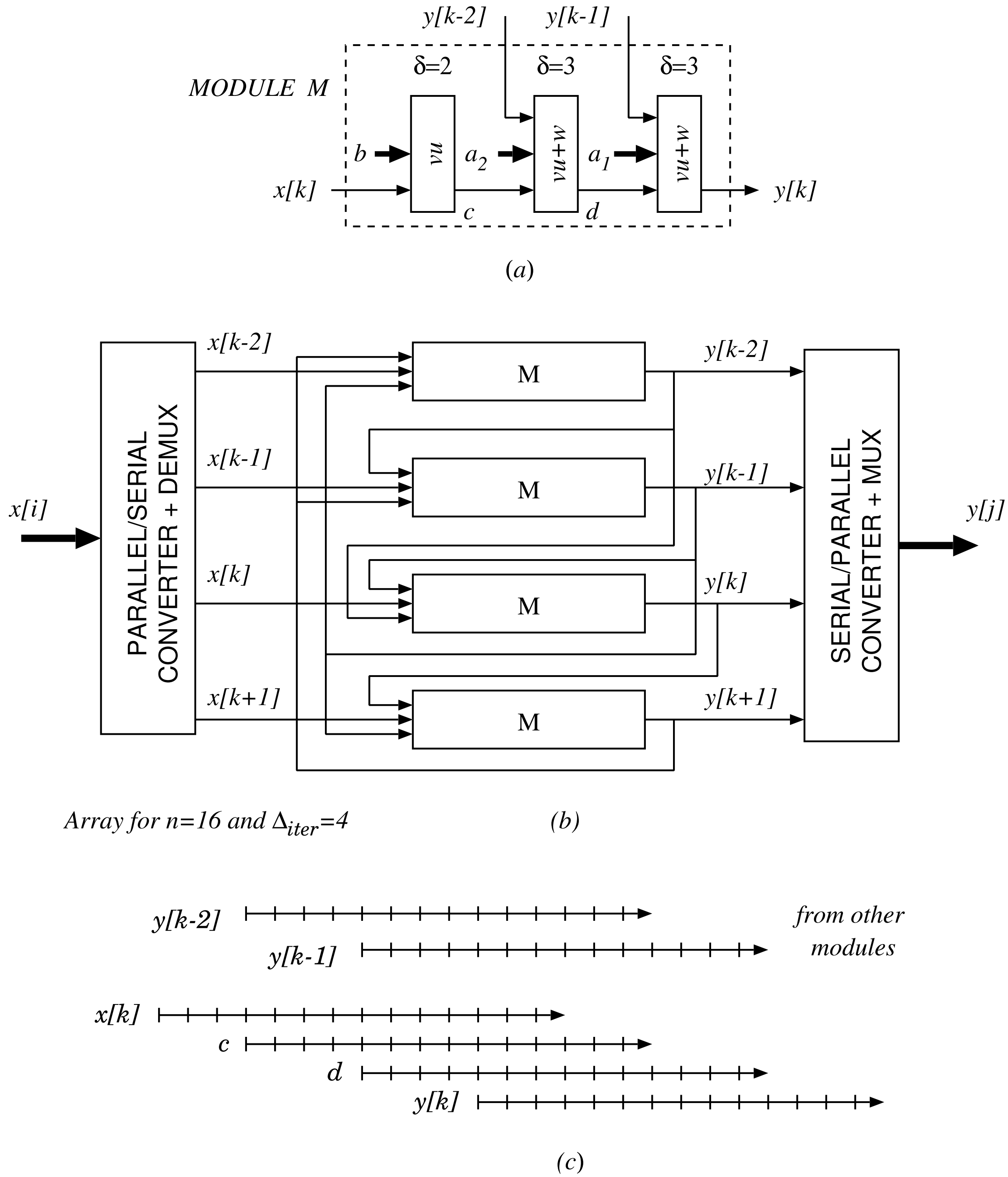


Figure 9.22: Online implementation of second-order IIR filter.