CH18 Circuits for Arithmetic Operations

Lecturer：呉安宇
Date：2006/1/6
Outline

18.1 Serial Adder with Accumulator
18.2 Design of a Parallel Multiplier
18.3 Design of a Binary Divider (Optional, Skipped)

Final hour:
Overview of Undergraduate Project
(by TA Chao)
Serial Addition Concept

- Similar to hand-operation by human
  - focus on 1 digit (bit) at one time $t_i$
- Use 1-bit full adder
  - Feed input bits $\{ x_i, y_i, c_i \}$ from LSB to MSB sequentially
  - Output sum bits from LSB to MSB sequentially
  - D F/F should be cleared before the operations

\[
\begin{array}{cccc}
  x_3 & x_2 & x_1 & x_0 \\
  y_3 & y_2 & y_1 & y_0 \\
  + & \text{Sum} & \text{Carry} \\
  s_3 & s_2 & s_1 & s_0 \\
  c_3 & c_2 & c_1 & c_0 \\
  t_3 & t_2 & t_1 & t_0 \\
\end{array}
\]
Timing of 4-bit Serial Adder with Accumulator (X: ACC, Y: Addend)

(a) At time $t_0$

(b) At time $t_1$

(c) At time $t_2$

(d) At time $t_3$

(e) At time $t_4$

final sum
Serial Adder with Accumulator

Verify the numerical example yourself

<table>
<thead>
<tr>
<th></th>
<th>X</th>
<th>Y</th>
<th>C_i</th>
<th>S_i</th>
<th>C_i+</th>
</tr>
</thead>
<tbody>
<tr>
<td>t_0</td>
<td>0101</td>
<td>0111</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t_1</td>
<td>0010</td>
<td>1011</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t_2</td>
<td>0001</td>
<td>1101</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t_3</td>
<td>1000</td>
<td>1110</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t_4</td>
<td>1100</td>
<td>0111</td>
<td>0</td>
<td>(1)</td>
<td>(0)</td>
</tr>
</tbody>
</table>
Serial Adder with Accumulator

\[ X = (x_3 x_2 x_1 x_0) \]
\[ Y = (y_3 y_2 y_1 y_0) \]

- SI: Serial Input
- Sh: Enable Signal of
- Shift registers (X & Y)
- Carry D F/F
Controller of Serial Adder with Accumulator

- Use Finite State Machine (FSM) for control unit of the serial adder
  - State graph and State table
Serial Adder with Accumulator

- Control Unit Design of Serial Adder
- Derivation of Control Circuit Equations

State Assignment

\[ D_A = A'B + AB' = A \oplus B \]

\[ D_B = St B' + AB' \]

\[ Sh = St + A + B \]
Serial Adder with Accumulator

Generalized Serial Arithmetic Circuit Design

FIGURE 18-5
Typical Serial Processing Unit

FIGURE 18-6
State Graphs for Serial Processing Unit

Feedback signals
Outline

18.1 Serial Adder with Accumulator
18.2 Design of a Parallel Multiplier
Concept of Shift-and-Add Multiplier

- Similar to hand-operation by human
  - Multiply 1 digit of multiplier to multiplicand
  - Shift Multiplier (x2) and add it to (+) the partial product

\[
\begin{align*}
\text{Multiplicand} & \rightarrow 1101 \quad \text{(13)} \\
\text{Multiplier} & \rightarrow 1011 \quad \text{(11)} \\
\text{Partial Products} & \rightarrow 1101 \\
\text{Product} & \rightarrow 10001111 \quad \text{(143)}
\end{align*}
\]
Share the Register of **Multiplier** and **Product**

{ product, multiplier }

- **Initial contents of product register**: 0 0 0 0 0 1 0 1 1 \(\rightarrow M\) (11)
- **(Add) (div by 2)**
- **After addition**:
  - 1 1 0 1 \(\rightarrow M\) (13)
  - 0 1 1 0 1 1 0 1 1
  - 0 0 1 1 0 1 1 0 1 \(\rightarrow M\)
- **After shift**:
  - 1 1 0 1
  - 1 0 0 1 1 1 1 0 \(\rightarrow M\)
  - 0 1 0 0 1 1 1 1 0
- **(Skip addition because \(M = 0\))**
- **After shift**:
  - 0 0 1 0 0 1 1 1 1 \(\rightarrow M\)
  - 0 0 1 0 0 1 1 1 1
- **(Add multiplicand because \(M = 1\))**
  - 1 1 0 1
  - 1 0 0 0 1 1 1 1 1
- **After addition**:
  - 1 0 0 0 1 1 1 1 1
  - 0 1 0 0 0 1 1 1 1 (143)

- **Dividing line between product and multiplier**
Structure of Serial Multiplier

Load: Load multiplier
Sh: Shift
Ad: Add

St: Start
M: multiply (LSB of Multiplier)
Done: finish multiplication
Implement FSM for Multiplier (1)

- Direct approach
  - Use states for each add/shift operation

Initial state

Load multiplicand and multiplier

1-bit multiplication:
- $S_1 \rightarrow S_2$: $M=1$, do addition
- $S_2 \rightarrow S_3$: do shift to Right
- $S_1 \rightarrow S_3$: $M=0$, do shift (addition is skipped)

Overhead to increase 1-bit multiplication: 2 states
Implement FSM for Multiplier (2)

- Separate approach
  - Use counter instead of state number for 1-bit multiplication
  - Complete signal $K$ is generate from counter

1-bit multiplication loop

$K'M'$: MSB of multiplier is 0, shift and goto $S_3$

$K$: Reach MSB and add operation is finished, shift and goto $S_3$
### Timing Table of Example

\[ \textstyle \begin{array}{cccccccc}
\text{Time} & \text{State} & \text{Counter} & \text{Product Register} & S_t & M & K & \text{Load} & \text{Ad} & \text{Sh} & \text{Done} \\
\hline
 t_0 & S_0 & 00 & 00000000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 t_1 & S_0 & 00 & 00000000 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
 t_2 & S_1 & 01 & 00000101 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
 t_3 & S_2 & 00 & 01101101 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 t_4 & S_1 & 01 & 00110110 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
 t_5 & S_2 & 01 & 10011101 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 t_6 & S_1 & 10 & 01001110 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
 t_7 & S_1 & 11 & 00100111 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
 t_8 & S_2 & 11 & 10001111 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\
 t_9 & S_3 & 00 & 01000111 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
\end{array} \]
Outline

18.1 Serial Adder with Accumulator
18.2 Design of a Parallel Multiplier
18.3 Design of a Binary Divider (Optional, Skipped)