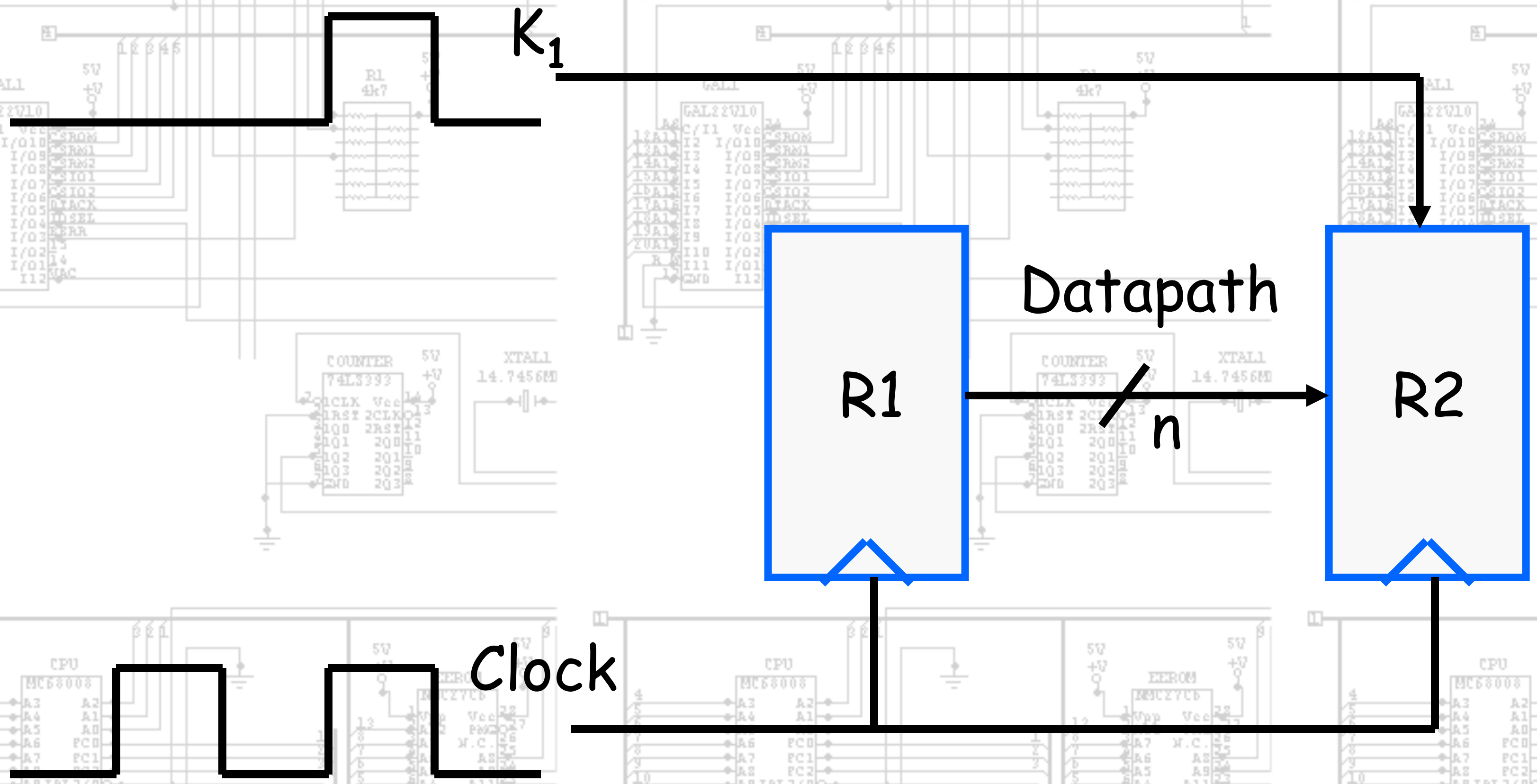
Register Transfer

- Describing large-scale processor activity.
- To discuss digital systems of this scale and level of complexity we need a number of descriptive tools.
- For example:

  a) Circuit schematics highlight the circuit components and their connectivity.

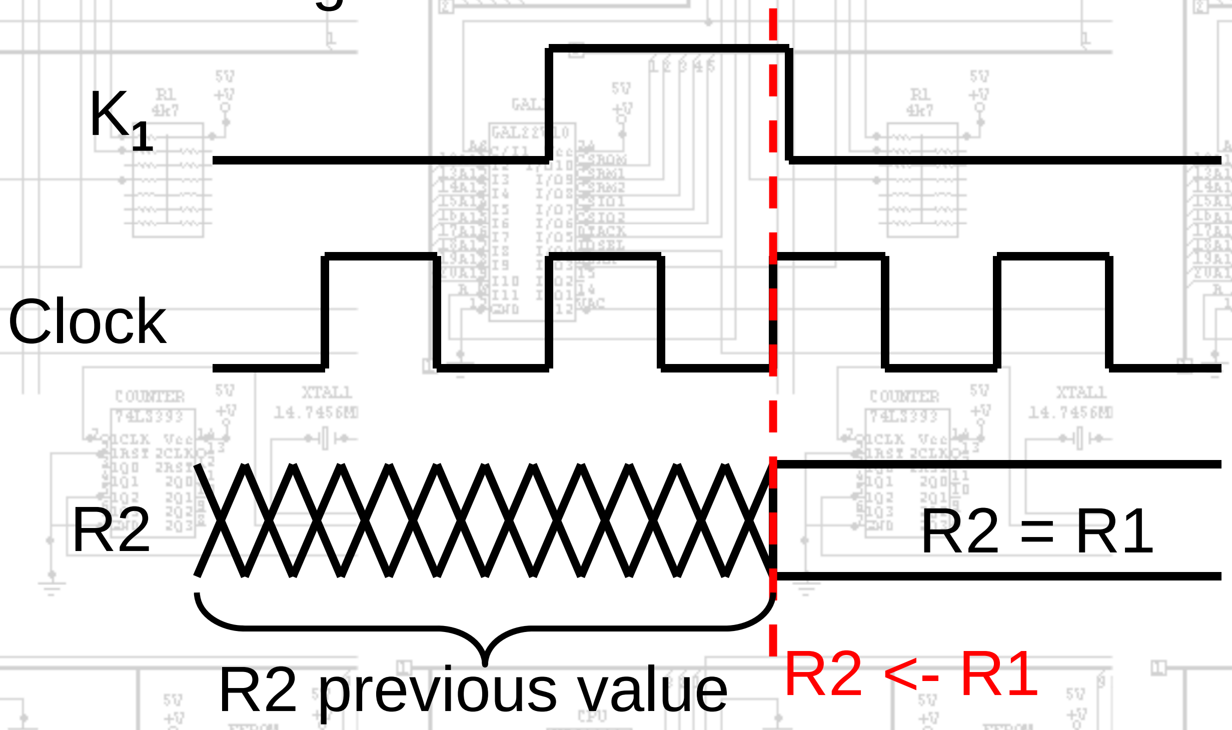
Transfer from R1 to R2 when K_1=1

Circuit Schematic



Timing Diagram

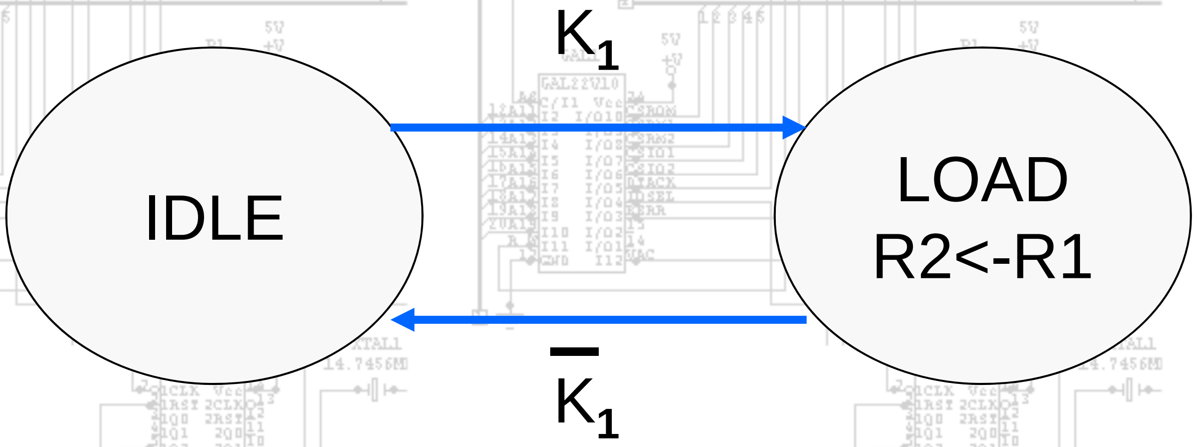
b) Timing diagrams highlight the detailed time sequence of transfer between registers.



The transfer R2 <-r1 occurs at the end of K_1

State Diagram

c) State diagrams highlight the modes of operation and their control



When the system is synchronous we normally omit the clock specification.

i.e. $K_1 \equiv K_1.\text{CLOCK}\uparrow$

Register Transfer Specification

- Source Register
- Destination Register
- Operation to be applied
- Condition or control function under which the transfer will occur.
  - We assume synchronous operation and omit the clock

$$\underbrace{K1}_{\text{Control Function}} : \overbrace{\underbrace{R2}_{\text{Destination Register}} <- \underbrace{R1}_{\text{Source Register}}}^{\text{Operation}}$$

Building Register-Transfer Statements

Symbol(s)	Description	Examples
Letters and Numerals	Denote Registers	AR, DR, R2, IR
Parentheses	Denote sections of Registers	R2(9), AR(2),R1(7:0)

Arrow	Denotes data transfer	R1<-R2
		IR<-DR

Comma	Separates simultaneous transfers	R1<-R2, R3<-AR

Square brackets	Denote memory addressing	DR<-M[AR]
		/* a read
		M[AR]<-DR
		/* a write

VHDL and RTL

Operation	RTL	VHDL
Combinational Assignment	=	<=(concurrent)
Register Transfer	$\leftarrow$	<=(concurrent)
Addition	+	+
Subtraction	-	-
Bitwise AND	^	and
Bitwise OR	$\vee$	or
Bitwise XOR	$\oplus$	xor
Bitwise NOR	$\overline{}$	not
Shift left (logical)	sl	sll
Shift right (logical)	sr	srl
Vector/Register	A(3:0)	A(3 downto 0)
Concatenation	||	&

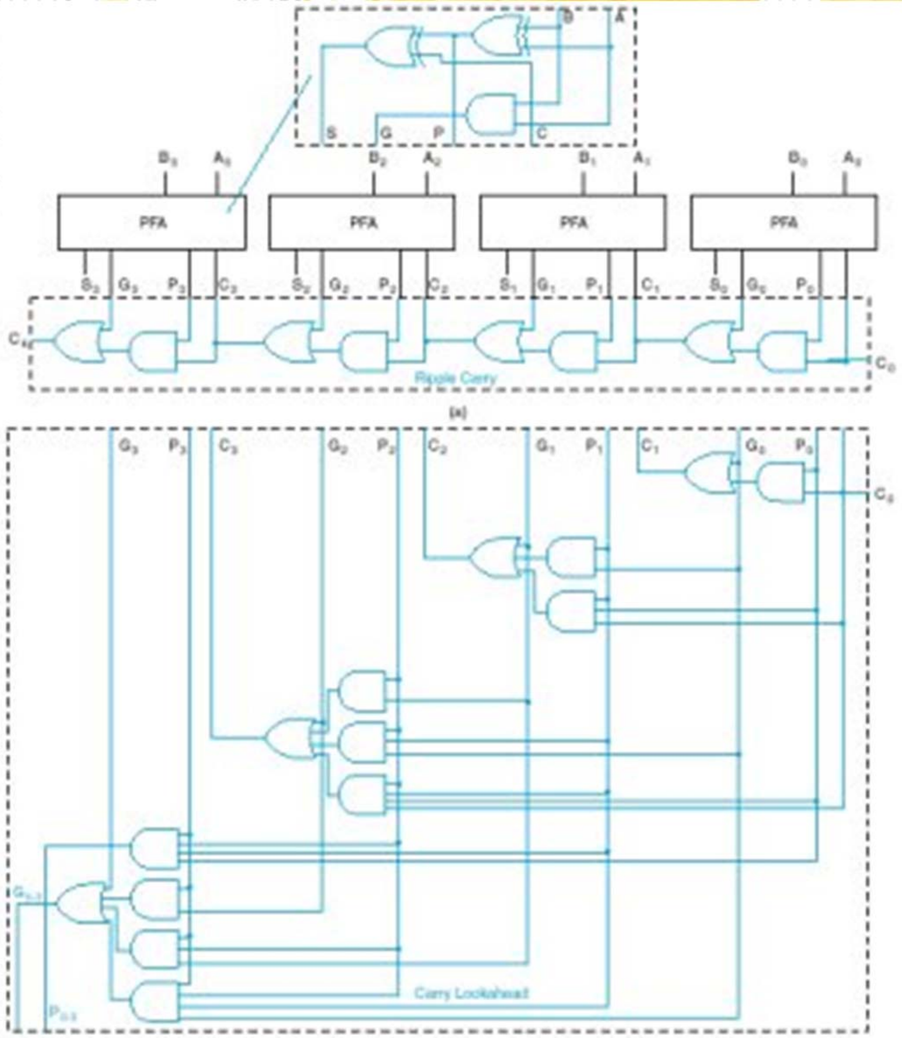
Micro-Operation

- A micro-operation is an operation which can be accomplished within a small number of gate propagation delays upon data stored in adjacent registers and memory.
- Those commonly encountered in digital systems divide naturally into four groups
  - Transfer or identity micro-ops copy data, e.g. R1<-R2, DR<-M[AR]
  - Arithmetic micro-ops provide the elements of arithmetic, e.g. R0<-R1+R2
  - Logic micro-0ps provide per bit opeartion, e.g. R1<-R2 or R2
  - Shift micro-ops provide bit rotations, e.g. R1<-sr R2, R0<- rol R1

Arithmetic Micro-ops

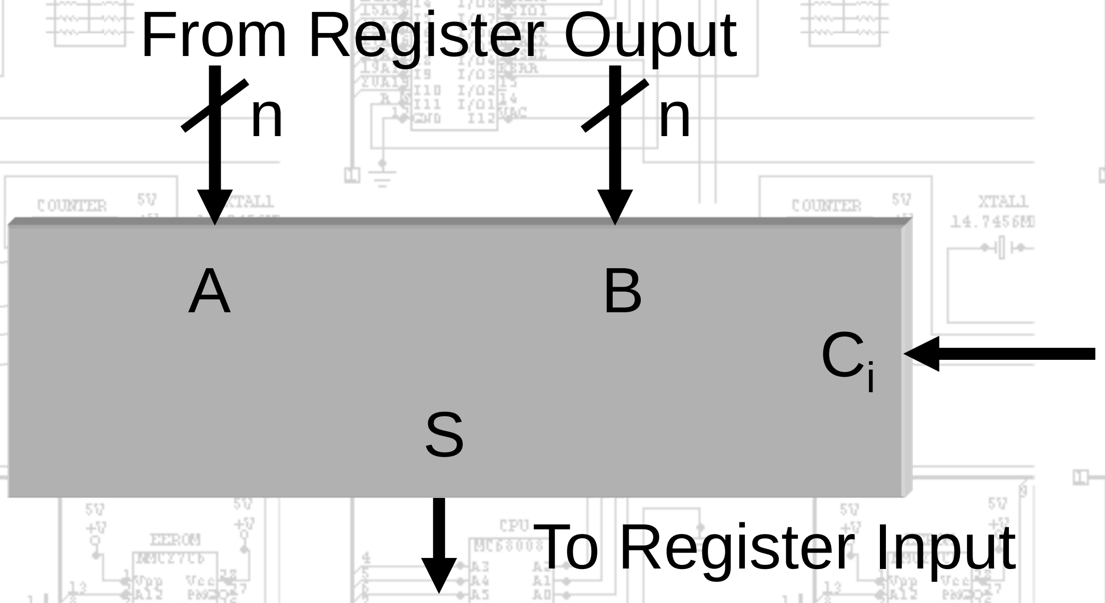
- These are operations which can be accomplished with a full-adder, which, with carry lookahead logic, can be made to deliver a substantial result, e.g. 64-bit in just a few gate delays.

Carry Lookahead Adder



CLA

- Let R0, R1,R3 be n-bit Register and consider what can be done with an n-bit CLA (carry lookahead adder)



Conditioned use of CLA

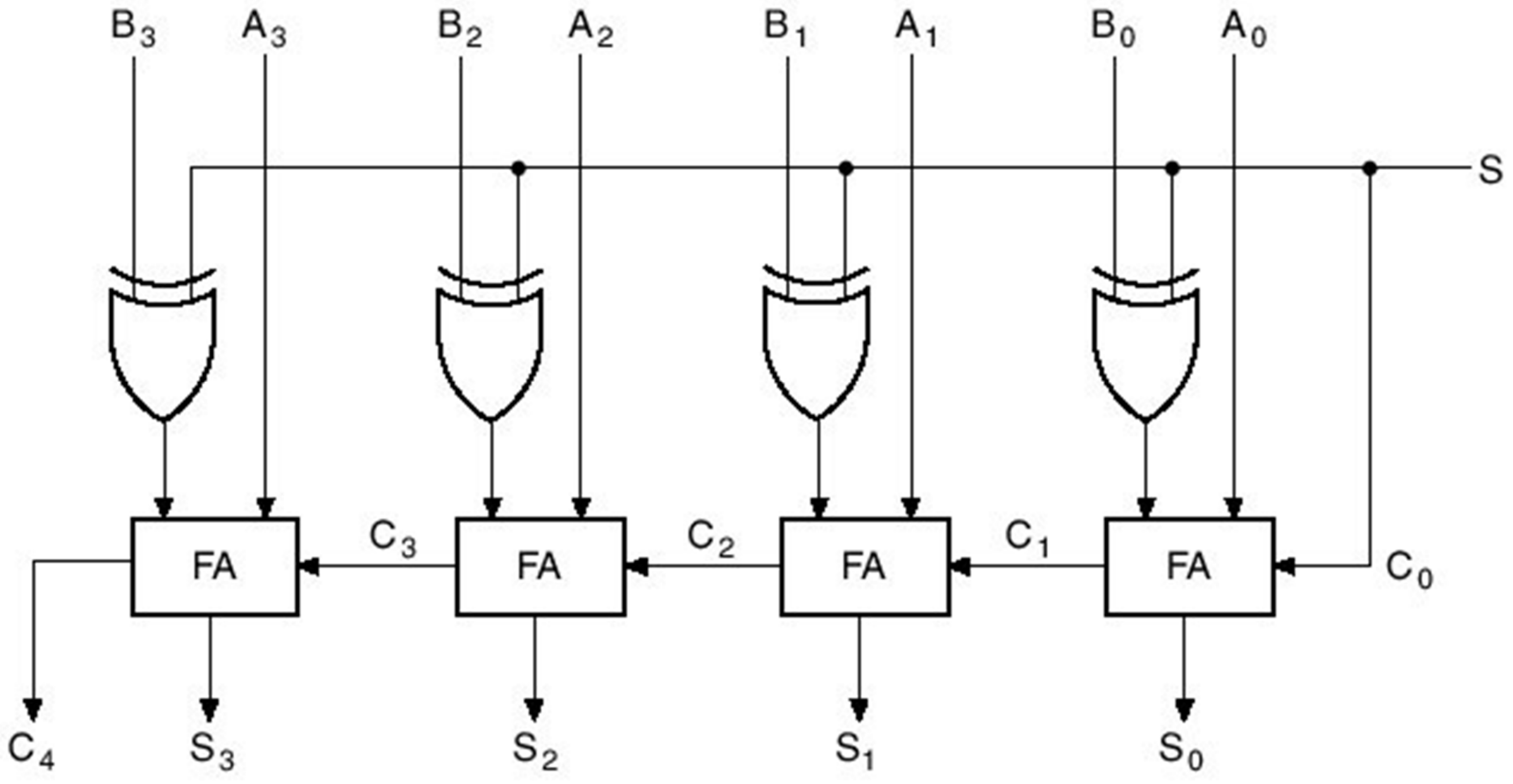
- By conditioning what arrives at A,B,C_i we can achieve:

Symbolic micro-op	CLA Inputs A		B		C	Function S
R0<-R1+R2	R1	+	R2	+	0	Addition
R0<-R1-R2	R1	+	$\overline{R2}$	+	1	Subtraction
R0<-R1+1	R1	+	0…0	+	1	Increment
R0<-R1-1	R1	+	1…1	+	0	Decrement
R0<-$\overline{R2}$	0…0	+	$\overline{R2}$	+	0	1's Complement
R0<-R2	0…0	+	$\overline{R2}$	+	1	2's Complement

Add &Sub Implementation

- The first two of these operations may be accomplished by the addition of an XOR gate to the B-input of each full-adder, as show on the next page.

Adder-Subtractor Circuit



B3
A3
B2
A2
B1
A1
B0
A0
S
FA
FA
FA
FA
C3
C2
C1
C0
C4
S3
S2
S1
S0