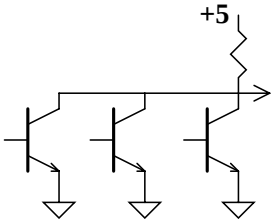


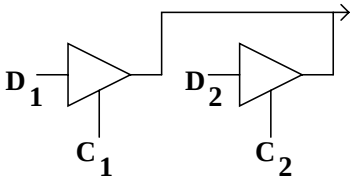

L4: Programmable Logic

Busses



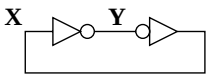
Open collector gates can be wired together like this to make wired ANDs.
This is a bus because it can be driven by more than one source.
You can't do this with Totem Pole outputs!

By controlling the gates on both transistors of a Totem Pole to be open, a high impedance is created (this is a tri−state). Control inputs C_1 and C_2 are output enables.



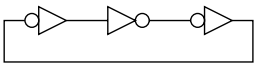
Feedback

Feedback produces 'State' which can be used to store information.



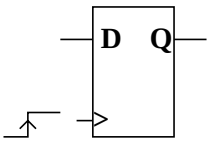
X	Y
0	1
1	0

Note that either state (X=0, Y=1 or X=1, Y−0) is valid.



What does this one do?

? **Try this in the lab.**



D	Q_n
0	0
1	1

Flip−flops are used to store state.

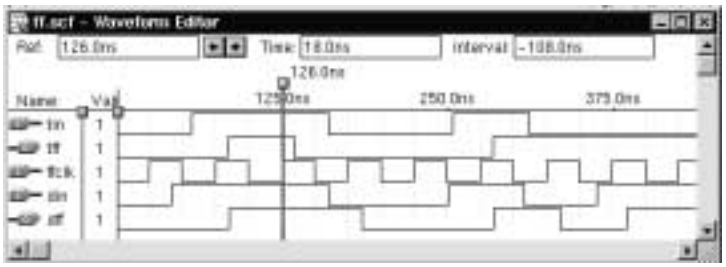
D Flip−Flop

Extract of Lab 1

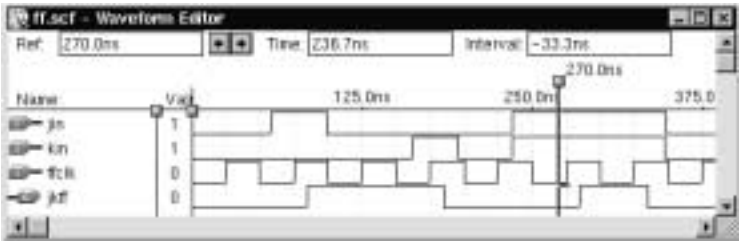
```
library ieee;
use ieee.std_logic_1164.all;
entity ff is
  port (ffclk, din, tin, jin, kin : in std_logic;
        dff, tff, jkff : out std_logic);
end ff;
architecture behavioral of ff is
  signal insidetff, insidejkff : std_logic;
begin
  process (ffclk) begin
    if rising_edge(ffclk) then
      dff <= din;
      insidetff <= tin xor insidetff;
      insidejkff <= (jin and (not insidejkff)) or ((not kin) and insidejkff);
    end if;
  end process;
  tff <= insidetff;
  jkff <= insidejkff;
end behavioral;
```

Simulation of the Extract

dff <= din; **tff <= tin xor tff;**



jkff <= (jin and (not jkff)) or ((not kin) and jkff);



Negative True and VHDL

- **Signals in VHDL are inherently positive true.**
  - **A signal is high (a one) when it "happens".**
- **A negative true signal is low (a zero) when it "happens".**
  - **Negative true signals are usually only used on I/O.**
  - **It is nice to be able to recognize whether a signal is negative or positive true from a clue provided by the signal name.**
  - **/foo is out as a VHDL identifier cannot begin with a slash.**
  - **nfoo could be troublesome for signal names starting with n.**
  - **not_foo and neg_true_foo are too verbose.**
- **By convention, we will prepend n_ to all signal names when the signal is negative true (low when it 'happens").**

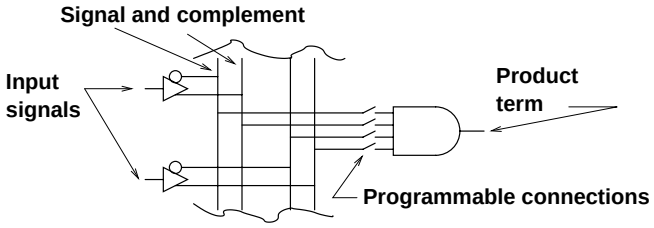
  N_foo

Negative or Positive True?

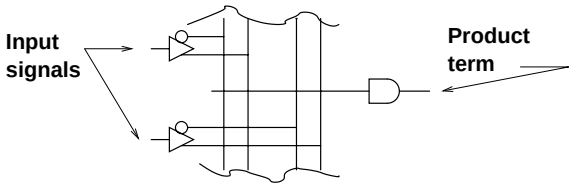
```
library ieee;
use ieee.std_logic_1164.all;
entity neg is
  port (a1, b1, a2, b2, a3, b3 : in std_logic;
        x, n_y, n_z : out std_logic);
end neg;
architecture equations of neg is
  signal y : std_logic;
begin
  -- The only clue we have are the signal names.
  -- The next two are positive true.
  x <= a1 AND b1;
  y <= a2 AND b2;
  -- The next two are negative true.
  n_y <= not y;
  n_z <= not (a3 OR b3);
end equations;
```


Programmable Logic

The basic element of all PALs (PLDs and CPLDs) is an AND gate which can be driven by each input and its complement. The unprogrammed state is that all connections are intact; therefore, the product term is zero. When all connections are destroyed (temporarily) the output of the AND floats high and is thus a one.

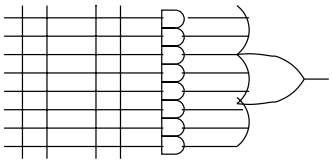


A shorthand notation (especially useful if there are a lot of inputs) is:



Or of Ands

The product terms produced by the ANDs are combined in large ORs.



And the output from the OR can usually be registered or simply . combinational. Additionally, either can be inverted or not.

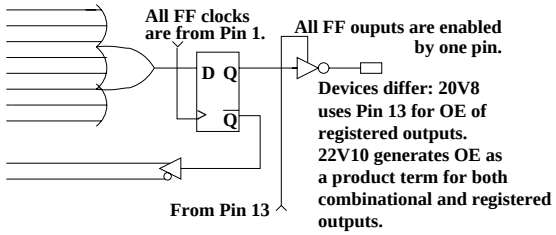
Pin	20V8(1)		20V8(2)		22V10	
	Use	Terms	Use	Terms	Use	Terms
23	I		I		I/O	8
22	O	7	I/O	8	I/O	10
21	I/O	7	I/O	8	I/O	12
20	I/O	7	I/O	8	I/O	14
19	I/O	7	I/O	8	I/O	16
18	I/O	7	I/O	8	I/O	16
17	I/O	7	I/O	8	I/O	14
16	I/O	7	I/O	8	I/O	12
15	O	7	I/O	8	I/O	10
14	I		I		I/O	8
13	I		/OE		I	

20V8 (1): All outputs are combinatorial (none registered).
(2): Some outputs registered: 8 product terms for registered outputs only.

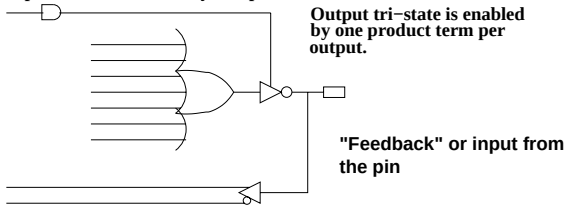
Outputs

This is how the 20V8 works.

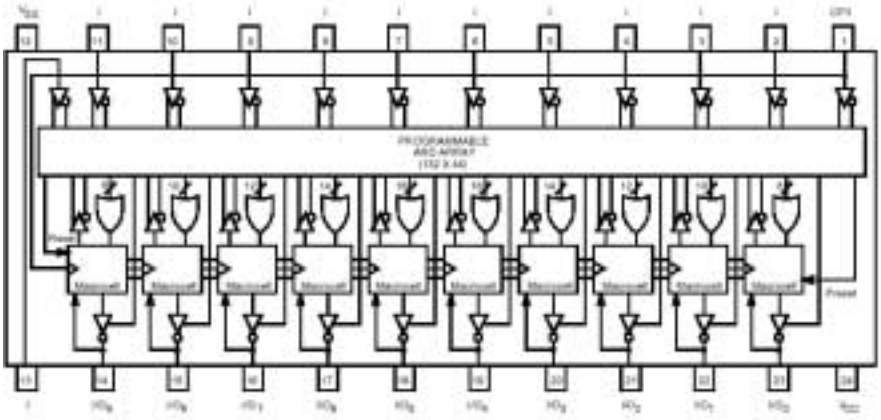
The output can be registered.



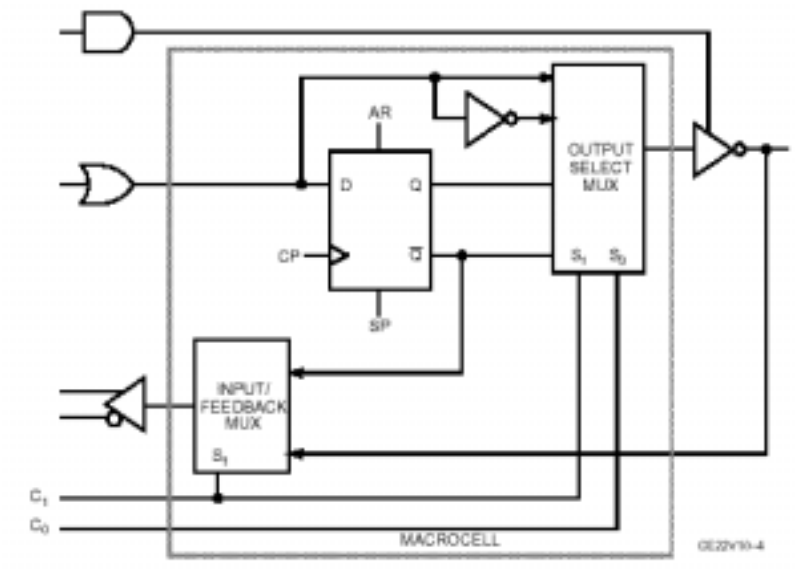
The output can be sent directly to a pin.



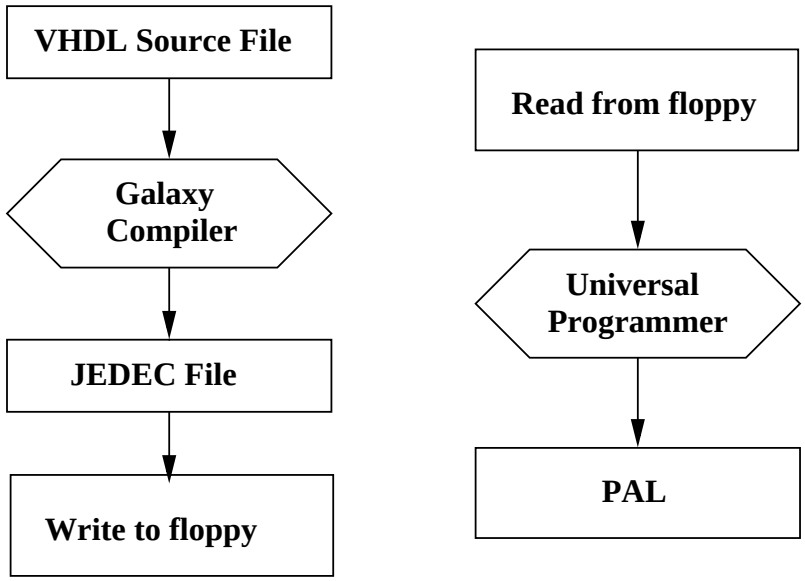
22v10 PAL



22V10 Macrocell



Programming Small PALs



Altera's FLEX 10K Devices

- Industry's first embedded PLDs
- Each FLEX device contains both embedded and logic arrays.
  - The embedded array consists of EABs.
    - Each of which provides 2,048 bits.
    - The EAB can be used to create RAM, ROM, dual-port RAM or a FIFO.
    - EABs can be used independently or they can be combined to implement larger logic functions.
  - The logic array consists of LABs.
  - LABs perform the same function as a sea-of-gates in a gate array.
    - They are used for general logic such as counters, adders, state machines, etc.
    - Each LAB consists of eight LEs and local interconnect.
    - An LE consists of a 4-input look-up table, a programmable FF, and dedicated signal paths for carry and cascade paths.

Altera's FLEX 10K Devices (more)

- **Based on reconfigurable CMOS SRAM elements**
  - **This enables 100% testing prior to shipment.**
  - **They can (must) be configured on their pc board.**
    - **This facilitates field changes and special test modes.**
- **Performance is shown on the next slide.**
  - **All performance values were obtained with Synopsis Designware or Library of Parameterized Functions (LPM).**
  - **They simply inferred or instantiated a function in Verilog, VHDL, AHDL, or a schematic.**
  - **Actually, one can mix design techniques in a single design.**
  - **These chips are now a few years old and they were not the top of the line when we acquired them.**
  - **The EPF10K10LC84-3 is the largest FLEX 10K FPGA that fits in an 84 pin plcc package.**
  - **The EPF10K70RC240-2 is the largest FLEX 10K FPGA that uses a five volt power supply. Larger ones use lower voltages which saves power (more on this later).**

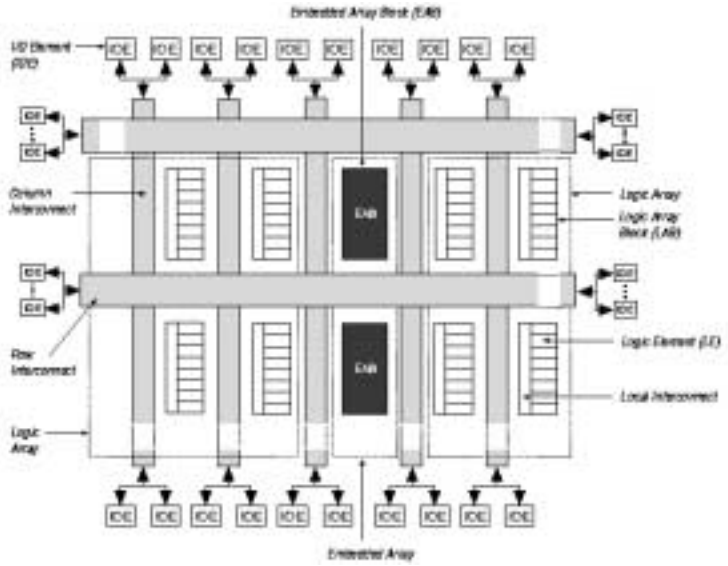
FLEX 10K and 10Ka Performance

Table 6. FLEX 10K & FLEX 10KA Performance

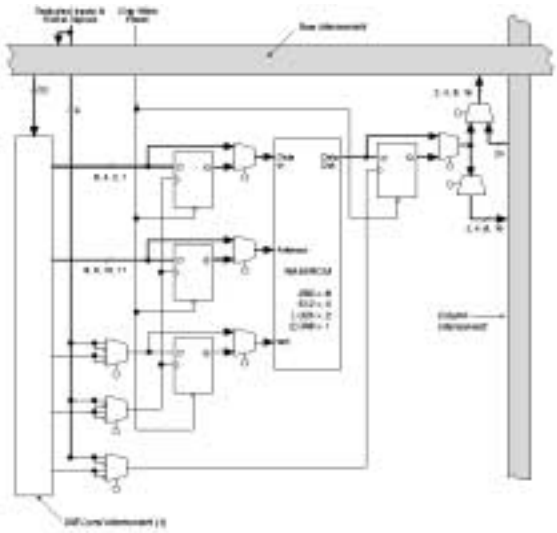
Application	Resources Used		Performance				Units
	LEs	EABs	-1 Speed Grade	-2 Speed Grade	-3 Speed Grade	-4 Speed Grade	
16-bit loadable counter (1)	16	0	204	166	125	95	MHz
16-bit accumulator (1)	16	0	204	166	125	95	MHz
16-to-1 multiplexer (2)	18	0	4.2	5.8	6.0	7.0	ns
256 × 8 RAM read cycle speed (3)	0	1	172	145	106	84	MHz
256 × 8 RAM write cycle speed (3)	0	1	106	88	88	63	MHz

Notes:
(1) The speed grade of this application is limited because of clock high and low specifications.
(2) This application uses combinatorial inputs and outputs.
(3) This application uses registered inputs and outputs.

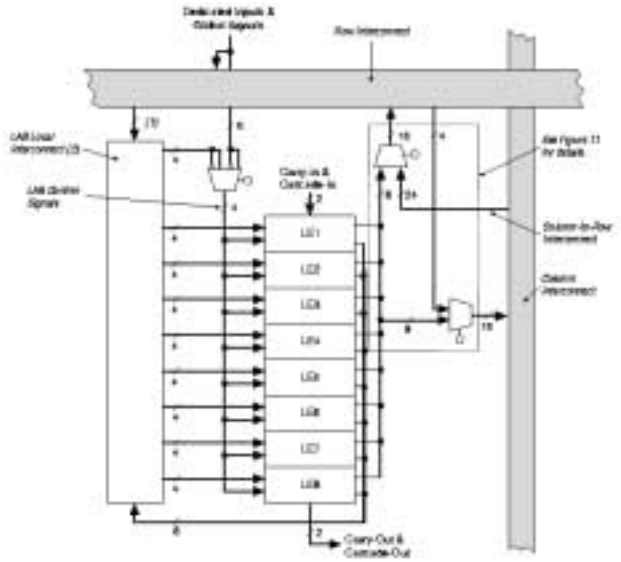
FLEX 10K Block Diagram



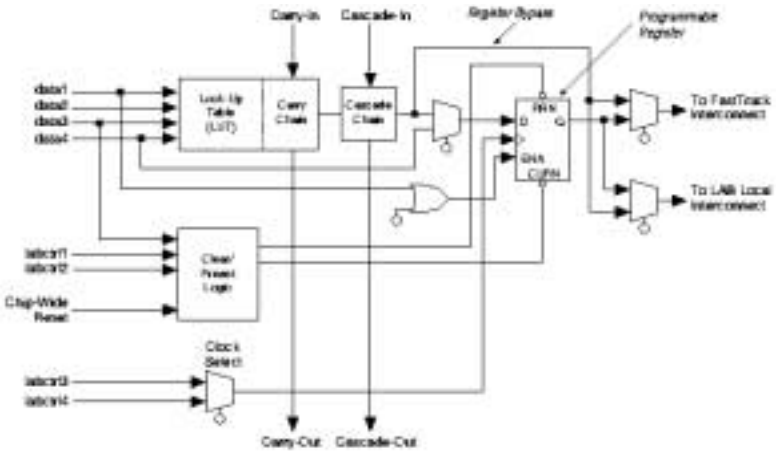
FLEX 10K Embedded Array Block



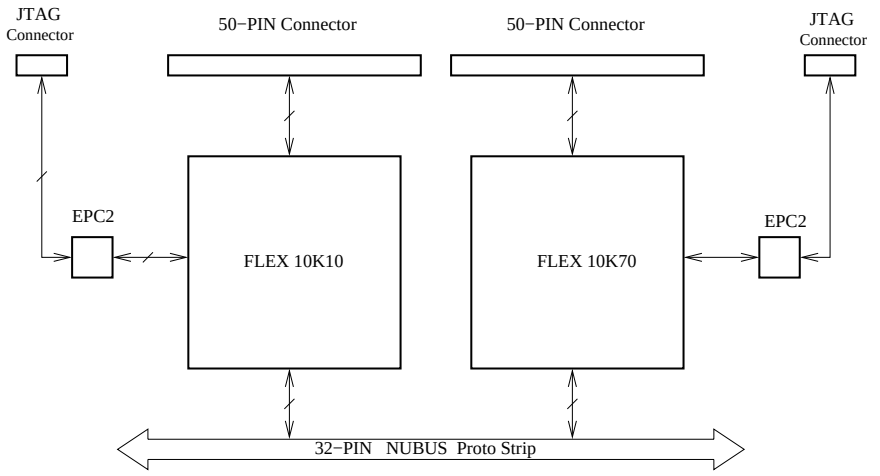
FLEX 10K Logic Array Block



FLEX 10K Logic Element



FPGA Module



FPGA Module

- Developed by Brian Perrin as his MEng thesis, April 2002
- You can connect to the FPGAs via the Nubus proto strip and by 50-pin cables to K1 or K1 of the kit.
  - Details are given in tables 1 and 2 of the handout.
  - You can also figure out connections between the FPGAs.
- BEWARE – Both CPLD and FPGA boards are interconnected via the nubus. Take this into account if you are using both.
- FPGAs are RAM based.
  - They are configured automatically when power is turned on as they are each wired to an EPC2 which is a serial flash prom.
  - MAX+PlusII is used to program the EPC2 with <project_name>.pof via a JTAG interface.
- "Erase" the EPC2s by programming with a file that tri-states each pin. See the handout.
- Ensure that there is no bus contention on any of the FPGA pins – on the nubus or 50-pin connectors.

MAX+PlusII

- MAX+PlusII runs on any Athena Sun workstation.
  - Type

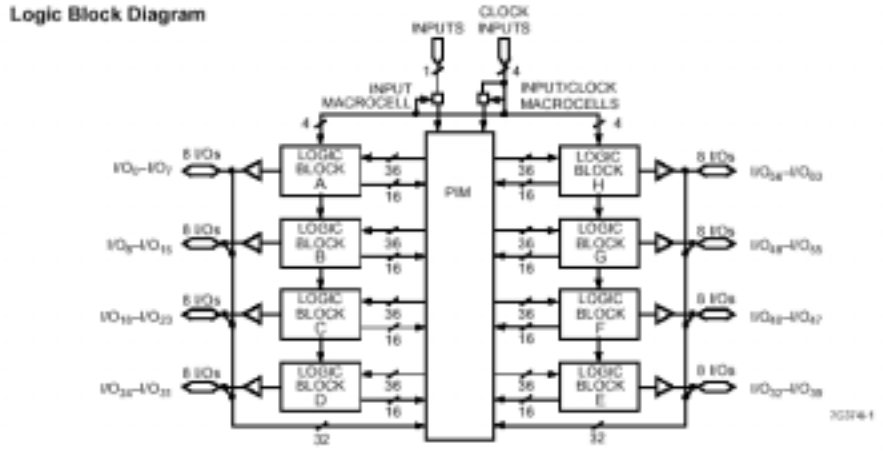
    setup 6.111
    max2win&
    - The first use of MAX+PlusII may cause fonts to be loaded. This should not happen on subsequent uses.
  - You may slogin to any of the ultra 5s in the digital lab.

    setup 6.111
    Ultra5 (This logs you in to the least lightly loaded Ultra5 in the lab.)
  - The PCs in the lab are not set up yet for vmware and MAX+PlusII.
    - We will announce when they are set up and post instructions on how to start up MAX+PlusII.
  - See information on the web page about MAX+PlusII.
    - In particular see
      - FPGA Module
      - Beginner's Guide to MAX+PlusII
      - MAX+Plus2 step by step

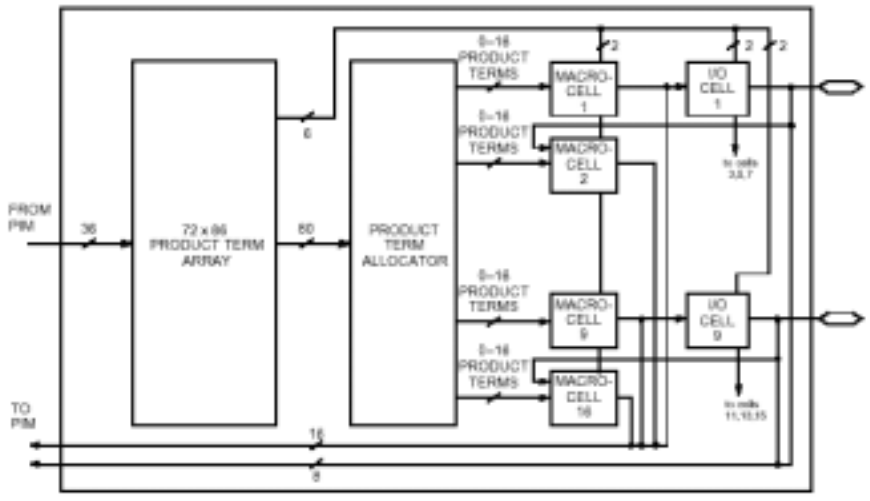
CY7C374i

- Manufacturer is Cypress
  - URL is http://www.cypress.com.
  - This contains info on 370 family and data sheet for 374i.
- CPLDs are just more complicated PLDs.
  - 128 macrocells in eight blocks
  - 64 I/O pins
  - 5 dedicated inputs including 4 clocks
    - Only one clock is usable on the 6.111 CPLD board.
  - JTAG interface – ISR – In-System Reprogrammable
  - High speed
    - Fmax = 125 Mhz
    - tPD = 10 ns
    - tS = 5.5 ns
    - tCO = 6.5ns

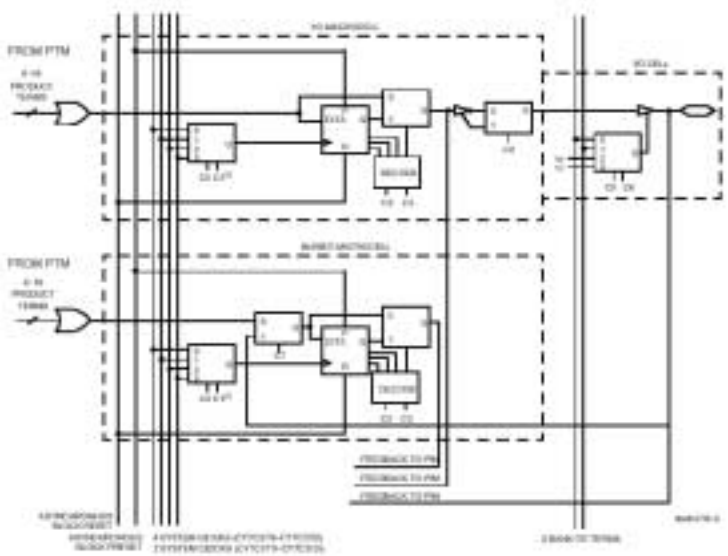
CY7C374i Block Diagram



CY7C374i Logic Block (1 of 8)

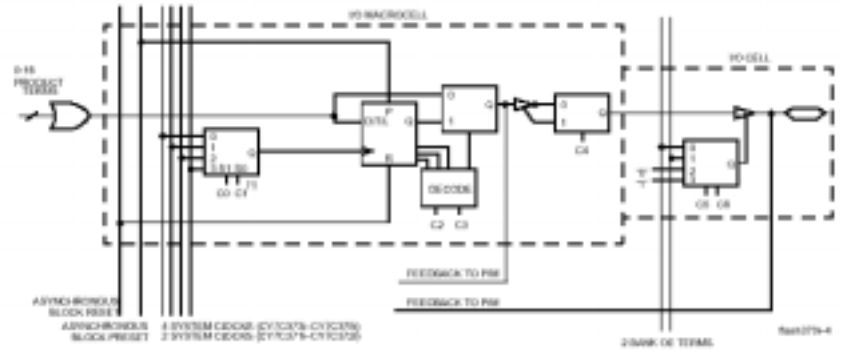


CY7C374i Macrocells

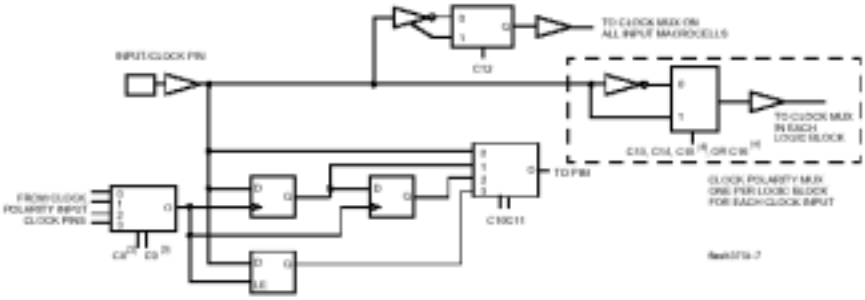


I/O in More Detail

- Note that this is similar to the 22V10.

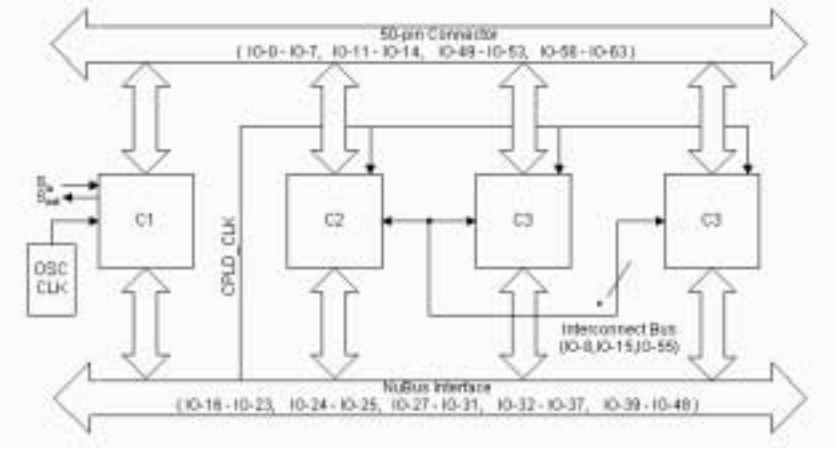


CY7C374i Input/Clock Pins



CPLD Board

Four Cypress 374i CPLDs which have 128 macrocells
The speed is 66Mhz. (Much faster than we can use!)
CPLD pins are accessible by the /AD bus and an optional 50-pin cable.
Interconnections between chips limit the flexibility of signal allocation.



CPLD Board (more)

BEWARE! Some signals accessible via the 50-pin cable are grounded on the kit. It depends on the 50-pin connector used on kit.

- **Three CPLDs are clocked by /AD31.**
- **The left CPLD can be optionally used as an RS-232 interface via the DB-9 connector.**

Jumpers select which CPLDs are to be programmed and the crystal for the left CPLD. This CPLD can also be clocked by a buffered version of the clock used for the right hand three CPLDs.

BEWARE – clocks must be disconnected during programming.