Percepatan Menggunakan Perangkat Keras

Pokok Bahasan:

FPGA, ASIC, CPLD

Tujuan Belajar:

Setelah mempelajari dalam bab ini, mahasiswa diharapkan mampu :

1. Mengetahui dan menjelaskan perkembangan penggunaan perangkat keras alternatif dalam sistem tertanam.
2. Mengetahui dan menjelaskan penggunaan perangkat keras alternatif dalam sistem tertanam untuk aplikasi : FPGA, ASIC, CPLD guna meningkatkan kinerja sistem tertanam tersebut.

14.1. Pendahuluan

Perangkat logika yang dapat diprogram (PLDs : Programmable Logic Devices) diperkenalkan pada pertengahan tahun 1970. Pemikiran dasarnya adalah memngkonstruksi kombinasi rangkaian logika yang dapat diprogram. Meskipun demikian, tidak seperti mikroprosesor ataupun mikrokontroller, yang dapat menjalankan program tetapi memiliki perangkat keras yang tetap, kemampuan pemrograman dari PLDs ditujukan pada **level perangkat keras**. Dengan kata lain PLD adalah *chip* bertujuan umum memiliki perangkat keras yang dapat diprogram untuk memenuhi spesifikasi tertentu.

PLDs yang pertama dikenal dengan PAL (Programmable Array Logic) atau PLA (Programmable Logic Array), tergantung pada skema pemrogramannya. Hanya menggunakan gerbang logika (tanpa *flip-flops*), dengan demikian hanya dimungkinkan penggunaannya untuk rangkaian kombinasional. Untuk mengatasi masalah ini, kemudian dikembangkan PLDs yang dilengkapi dengan *register*, yang meliputi satu flip-flop untuk setiap keluaran dari rangkaian. Untuk itu, rangkaian fungsi *sequential* dapat diimplementasikan dengan baik.

Pada awal tahun 1980, rangkaian logika tambahan diterapkan untuk setiap keluaran PLD. Sel baru ini disebut *Macrocell*, berisikan (disamping *flipflops*) gerbang logika dan *multiplexer*. Dan lebihnya lagi, sel tersebut dapat diprogram, mengijinkan beberapa mode dari pengoperasiannya. Dan lagi, sel ini memberikan kemampuan pengembalian sinyal (*feedback*) dari keluaran ke rangkaian array terprogram, sehingga meningkatkan fleksibilitas dari PLD. Struktur baru dari PLD ini dikenal sebagai *generic* PAL (GAL).Arsitektur sejenis diketahui sebagai perangkat PALCE (PAL CMOS Electrically erasable/programable). Keseluruhan *chips* ini (PAL, PLA, resgitered PLD, dan GAL/PALCE) termasuk kategori SPLDs (Simple PLDs).

Beberapa perangkat GAL difabrikasi dalam chip yang sama, menggunakan skema routing yang canggih, teknologi silicon terbaru, dan penambahan beberapa ciri-ciri khusus (seperti JTAG Support dan antarmuka dengan beberapa gerbang logika standar). Pendekatan ini dikenal kemudian sebagai CPLD (Complex PLD) dan menjadi populer dikarenakan kepadatan yang tinggi, kinerja yang tinggi, dan murah.

Akhirnya, pada pertengahan 1980, FPGAs (Field Programmable Gate Arrays) diperkanalkan. Berbeda dari CPLD untuk arsitektur, teknologi, ciriciri khusus yang sudah ada, dan harga. Dan ditujukan untuk implementasi pada skala ukuran yang besar dan kinerja rangkaian yang tinggi. Tabel 14. 1 memaparkan ringkasan dari evolusi PLD.

Tabel 14. Ringkasan dari evolusi PLD.

<table>
<tr><td rowspan="3">PLDs</td><td rowspan="2">Simple PLD (SPLD)</td><td>PAL
PLA
Resgitered PAL/PLA
GAL</td></tr>
<tr><td colspan="2">Complex PLD (CPLD)</td></tr>
<tr><td colspan="2">FPGA</td></tr>
</table>

Catatan : seluruh PLDs (simple ataupun complex) adalah *non-volatile*. Mereka dapat diprogram secara OTP (*one-time programable*), dengan menggunakan teknik *fuses* atau *antifuses*, atau dapat diprogram kembali, dengan EEPROM atau memori Flash (Flash adalah teknolgi terpilih dalam perangkat baru sekaran ini) . FPGA, sebaliknya adalah

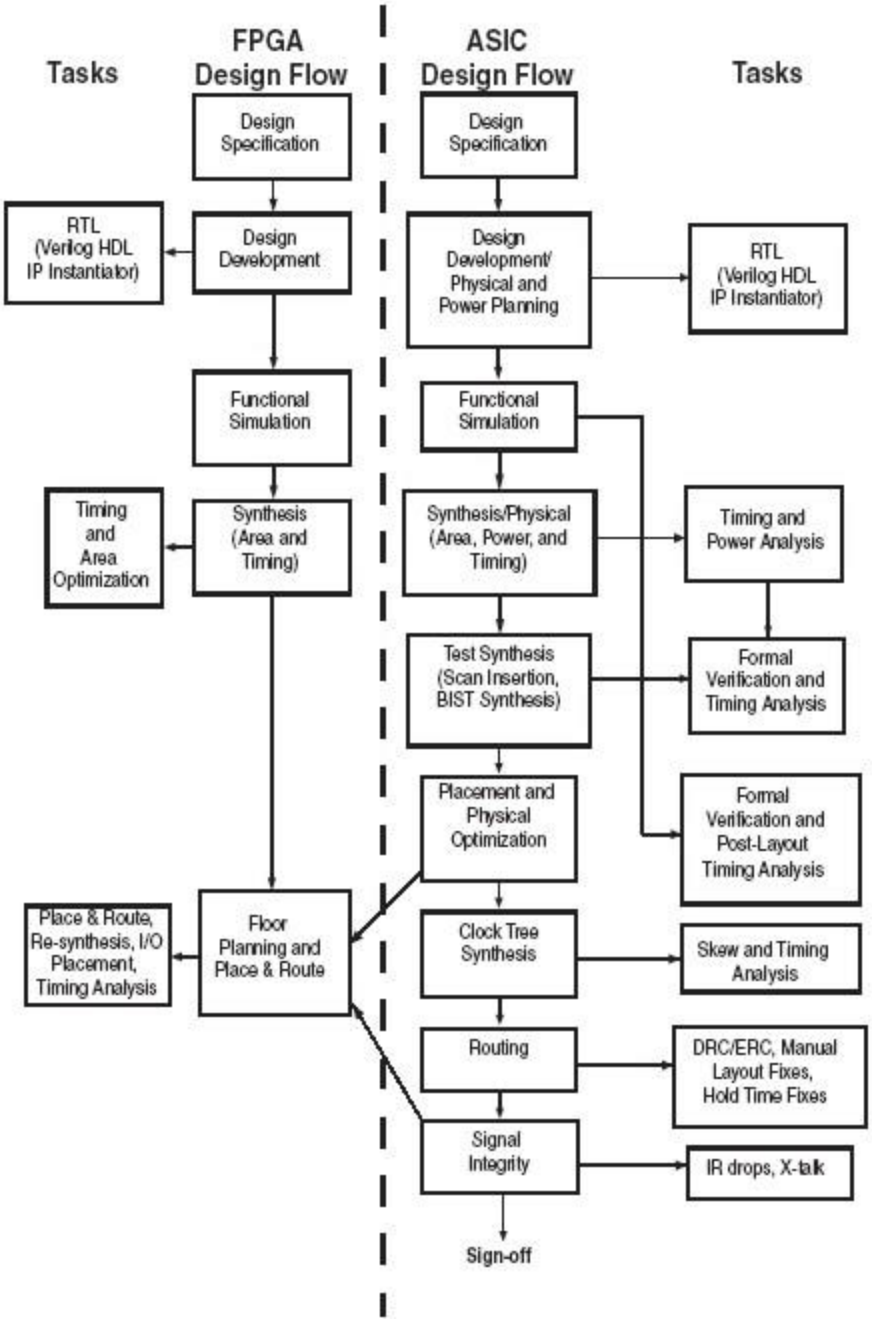
sangat *volatile*, untuk itu digunakan SRAM untuk menyimpan susunan koneksi, yakni konfigurasi ROM diperlukan untuk mengambil susunan koneksi pada saat dihidupkan (*power up*). Meskipun demikian, ada teknik untuk membuat FPGA *non-volatile* seperti digunakannya teknik antifuse.

Pemrograman FPGA dapat menggunakan bahasa pemrograman tingkat tinggi yakni Verilog atau VHDL (VHSIC Hardware Description Language), dan VHSIC sendiri adalah Very High Speed Integration Circuit. VHDL ditujukan untuk sintesis dan simulasi rangkaian (*synthesis* dan *simulation*). Meskipun demikian, VHDL dapat disimulasikan sepenuhnya, tidak seluruh konstruksi rangkaian dapat disimulasikan. Motivasi dasar penggunaan VHDL dibandingkan kompetitornya Verilog, bahwa VHDL adalah standar, dan bahasa yang tidak tergantung pada teknologi maupun vendornya, sehingga dapat dipindahkan (*portable*) dan dapat digunakan kembali (*reusable*). Dua aplikasi utama VHDL adalah dalam bidang Programmable Logic Device (CPLDs dan FGPAs) dan dalam bidang ASICs (Application Specific Integrated Circuits). Sekali kode VHDL ditulis, dia dapat digunakan untuk implementasi rangkaian dalam perangkat terprogram (baik dari Altera, Xilinx, Atmel, dll) atau dapat dikrimkan ke untuk difabrikasi sebagai *chip* ASIC. Saat ini banyak *chip* komersial (mikrokontroler misalnya) dirancang menggunakan pendekatan ini.

Perlu diperhatikan bahwa tidak seperti program komputer pada umumnya, yang diproses secara berurutan (*sequential*), perintah dalam VHDL secara inheren adalah paralel (*concurrent*). Dengan alasan itu VHDL lebih dikenal sebagai kode daripada program. Pada VHDL hanya perintahperintah yang ditempatkan dalam PROCESS, FUNCTION atau PROCEDURE akan dijalankan secara sekuensial atau berurutan. Gambar 4.1 menyajikan perbandingan pengembangan berdasarkan ASIC dan FPGA sedangkan gambar 14.2. menyajikan aliran rancangan modul/komponen berbasiskan FPGA menggunakan VHDL.

Seperti telah disampaikan bahwa salah satu utilitas utama dari VHDL adalah kemampuan sintesis rangkaian/sirkuit atau sistem dalam perangkat terprogram (PLD atau FPGA) atau di dalam suatu ASIC. Rancangan dimulai dengan menuliskan kode VHDL dan dismpan dalam file berekstensi .vhd sama dengan nama ENTITY-nya. Langkah pertama dalam sintesis adalah proses kompilasi, yakni mengkonversi pemrograman tingkat tinggi menggunakan VHDL, yakni deskripsi dari rangkaian pada level RTL (Register Transfer Level) ke dalam *netlist* pada level gerbang (Gate Level). Langkah ke dua adalah optimisasi, yang dilakukan pada level gerbang untuk kecepatan atau area. Pada tahap ini, rancangan dapat disimulasikan. Akhirnya perangkat lunak

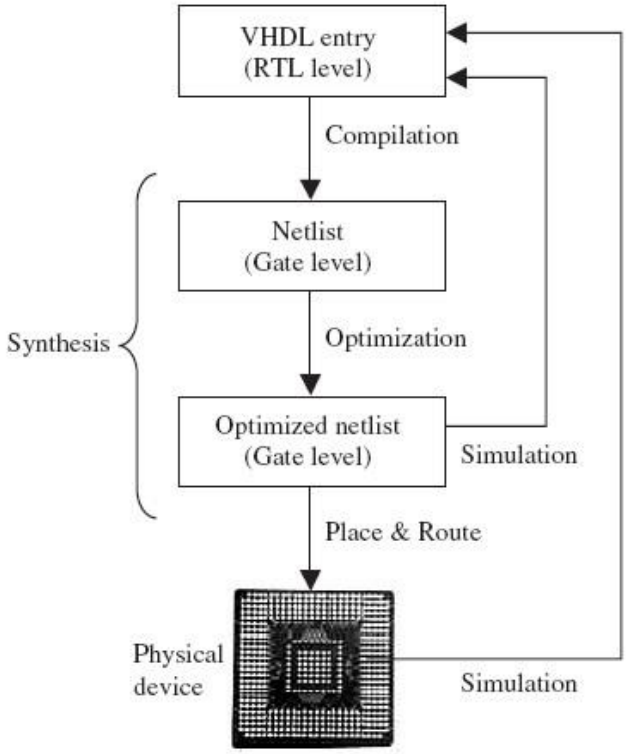
penempatan (*fitter, place and route*) akan membangun layout fisik untuk chip PLD/FPGA atau membentuk tanda (*masks*) untuk ASIC.



Gambar. 14.1. Perbandingan aliran pengembangan FPGA dan ASIC

Proses rancangan tersebut pada umumnya menggunakan alat bantu perangkat lunak EDA (Electronic Design Automation), terdapat beberapa perangkat lunak untuk sintesis, implementasi dan simulasi menggunakan VHDL. Beberapa alat bantu

ditawarkan merupakan bagian dari ruang lingkup rancangan dari vendor (seperti Altera's Quartus II untuk sintesis kode VHDL ke chip CPLD/FPGA Altera, atau Xilinx ISE untuk CPLD/FPGA dari Xilinx).



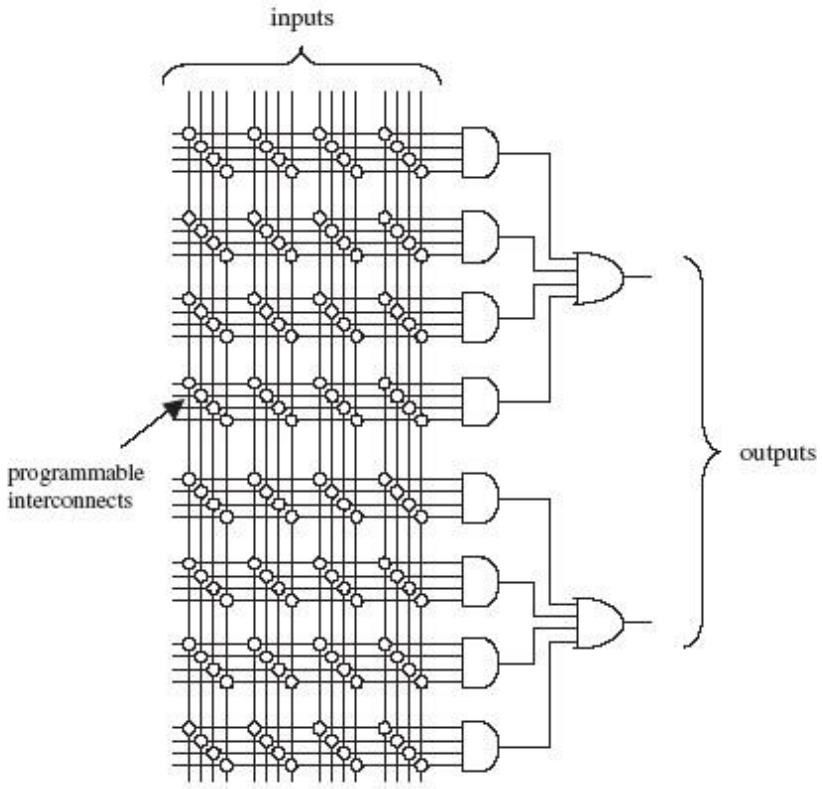
Gambar. 14.2. Aliran Rancangan Perangkat Menggunakan VHDL

14.1.1. SPLD

PAL (Programable Array Logic) diperkenalkan oleh Monolithic Memories pada pertengahan tahun 1970. Arsitektur dasarnya diilustrasikan secara simbol dalam gambar 14.3, dimana lingkaran adalah koneksi yang dapat diprogram. Tampak bahwa rangkaian ini merupakan komposisi aray terprogram dari gerbang AND, diikuti larik tetap gerbang OR. Implementasi dari gambar 14.2 didasarkan pada kenyataan bahwa setiap fungsi kombinasional dapat diwakili oleh Sum-of-Product (SOP); yakni beberapa a_1, a_1, a_2, ..., a_N adalah masukan logika, maka kombinasi keluaran x dapat dihitung sebagai :

$$X = m_1 + m_2 + \ldots + m_M ,$$

Dimana mi=$fi(a_1, a_1, a_2, ..., a_N)$ adalah minterm dari fungsi x.



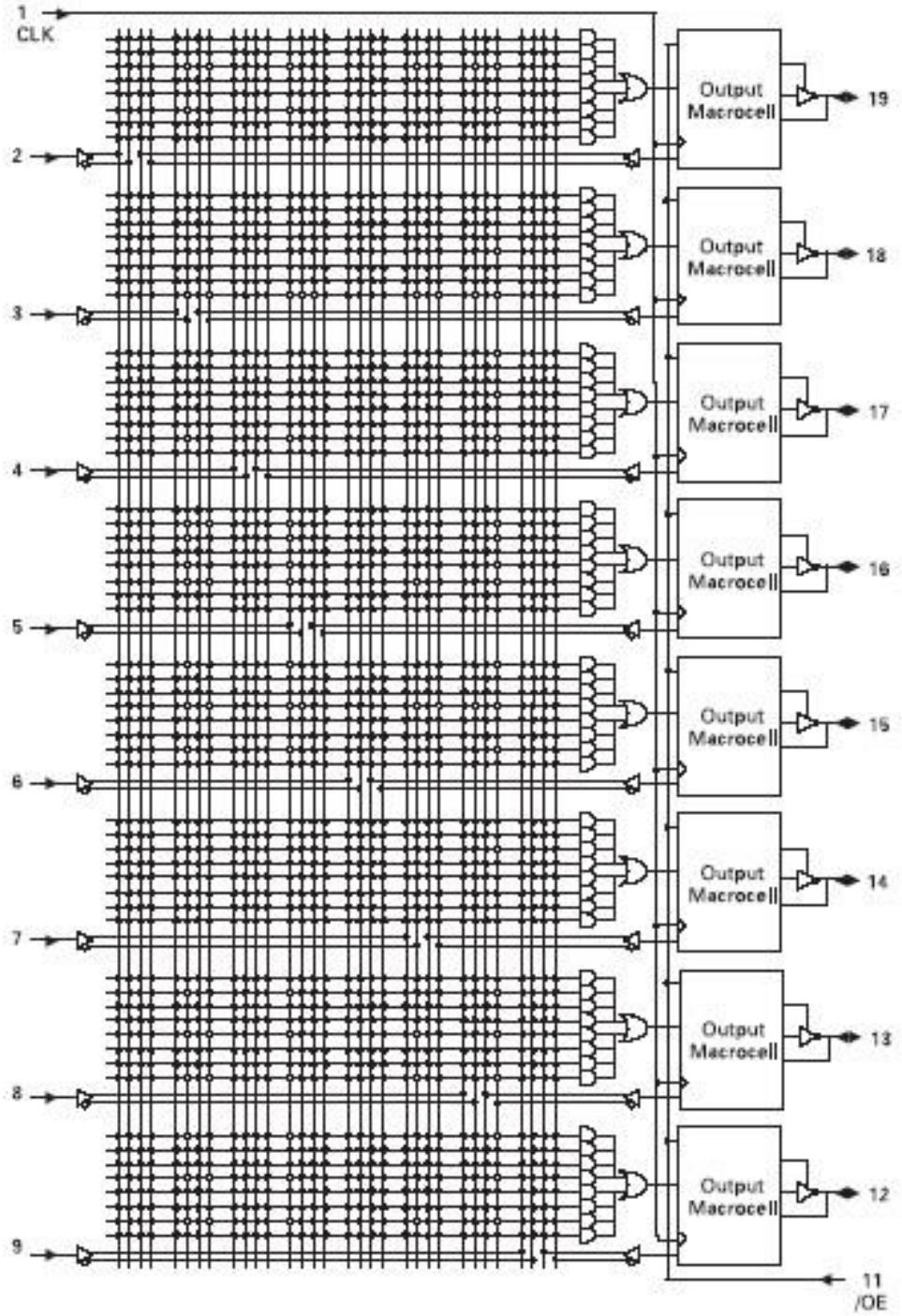
Gambar 14.3. Ilustrasi arsitektur PAL.

Dengan demikian hasil keluaran (minterm) dapat diperoleh dari gerbang AND yang keluaranya dihubungkan kepada gerbang OR yang akan menghitung keseluruhannya (sum) sebagai implementasi SOP.

Salah satu SPLD lainnya yakni GAL (Generic PAL), arsitekturnya diperkenalkan oleh Lattice pada awal tahu 1980. Berisikan beberapa peningkatan dari pada PAL sebelumnya : pertama, sel keluaran yang lebih canggih (Macrocell) dikonstruksikan, yang berisikan *flip-flop* dan beberapa gerbang *multiplexer*; kedua, Macrocell itu sendiri dapat diprogram, memungkinkan beberap mode operasi; ketiga, pengembalian sinyal dari keluaran Macrocell ke larik yang dapat diprogram dapat dilakukan, membuat rangkaian lebih fleksibel; keempat, EEPROM digunakan dari pada PROM atau EPROM.

Gambar 14.4 menunjukkan contoh dari perangkat GAL, GAL 16V8 (V = Versatile). Perangkat ini memiliki 16 masukan, 8 keluaran dalam paket 20pin. Konfigurasi aktual adalah 8 pin masukan (pin 2-9) dan 8 IN/OUT pin (pin 12 –19), CLK (pin 1), OE (-Output

Enable, pin 11), VDD (pin 20) dan Ground (pin 10). Pada setiap keluaran terdapat Macrocell (setelah gerbang OR). Interkoneksi yang dapat diprogram digambarkan dengan lingkaran kecil.



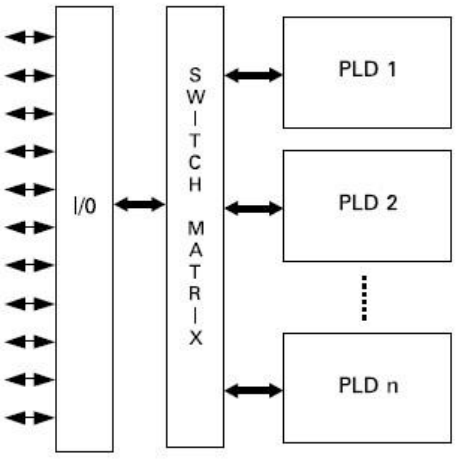
Gambar 14.4. Arsitektur GAL 16V8.

Perangkat GAL saat ini menggunakan teknologi CMOS, 3.3 V, EEPROM atau teknologi Flash, dan maksimum frekwensi sekitar 250 MHz. Beberapa perusahaan diaantaranya : Lattice, Atmel, TI, dll).

14.1.2. CPLD

Pendekatan dasar untuk CPLD disajikan dalam gambar 14.3, tampak bahwa beberapa PLDs (pada umumnya adalah GAL) difabrikasi dalam chip tunggal, dengan susunan sakelar matriks terprogram menghubungkan semuanya dan ke kaki I/O. Selebihnya, CPLD pada umumnya memiliki sedikit ciri-ciri khusus tambahan, seperti dukungan JTAG dan antarmuka ke rangkaian logika standar (1.8 V, 2.5 V, 5 V, dll).

Berkaitan dengan gambar 14.5, sebagai contoh Xilinx XC9500 CPLD, berisikan n PLDs, masing-masing mewakili perangkat 36V18 GAL, dimana n = 2, 4, 6,8, 12 atau 16. Beberapa perusahaan pembuat CPLDS, seperti Altera, Xilinx, Lattice, Atmel, Cypress, dll. Sebagai contoh dari dua perusahaan Altera dan Xilinx diilustrasikan pada tabel 14.2 dan 14.3. Dapat dilihat lebih dari 500 macrocells dan lebih dari 10,000 gerbang terlihat pada tabel tersebut.



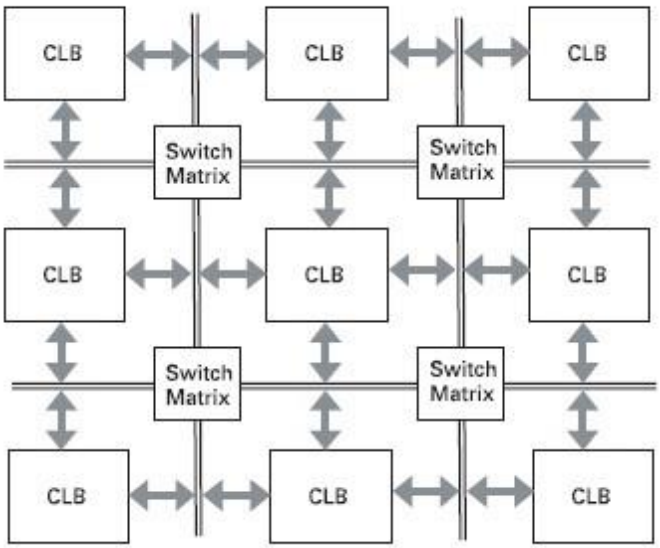
Gambar 14.5. Arsitektur CPLD

14.1.3. FPGA

Field Programmable Gate Array diperkenalkan oleh Xilinx pada pertengahan tahun 1980. Arsitektur dasar dari FPGA disajikan pada gambar 14.6, berisikan matrik CLB (Configurable Logic Block), saling dihubungkan dengan sederetan matrik saklar.

Arsitektur dalam dari CLB (gambar 14.5) berbeda berbeda dari PLD (gambar 14.4). Yang pertama, implementasi ekspresi SOP tidak menggunakan gerbang AND yang diikuti oleh gerbang OR seperti pada SPLD, tetapi operasi itu didasarkan pada LUT (look up table).Kelebihannya di dalam

FPGA jumlah flip-flop menjadi sangat banyak dari pada CPLD, dan memungkinkan konstruksi rangkaian sekuensial yang lebih kompleks. Dukungan JTAG dan antarmuka ke level logika yang berbeda, tambahan ciri khusus lainnya adalah : SRAM memori, multiplikasi clock (PLL atau DLL), antarmuka PCI, dll. Beberapa *chip* juga dilengkapi dengan blok terdedikasi, seperti : multipliers, DSP dan mikroprosessor.



Gambar 14.6. Arsitektur Dasar FPGA.

Perbedaan dasar lainnya antara FPGA dan CPLD mengacu pada interkoneksi penyimpanan. CPLD digunakan non-volatile (penggunaan antifuse, EEPROM, Flash,dll), pada umumnya FPGA menggunakan SRAM, yang berarti volatile; Pendekatan ini menghemat ruang dan menurunkan biaya pembuatan chip, dikarenakan FPGA menyajikan sejumlah besar interkoneksi yang dapat diprogram, meski memerlukan

ROM eksternal. Meskipun demikian terdapat FPGA yang non-volatile (dengan antifuse), dimana memberikan keuntungan apabila pemrogramman ulang tidak perlu dilakukan.

Table. 14.2. CPLDs dari Altera.

Family	*Max7000 (B, AE, S)*	*MAX3000 (A)*	*MAX II (G)*
Macrocells/ LUTs	32–512 macrocells	32–512 macrocells	240–2,210 LUTs (192–1,700 equiv. macrocells)
System gates	600–10,000	600–10,000	
I/O pins	32–512	34–208	80–272
Max. internal clock freq.	303 MHz	227 MHz	304 MHz (I/O limited)
Supply voltage	2.5 V (B), 3.3 V (AE), 5 V (S)	3.3 V	1.8 V (G), 2.5 V, 3.3 V
Interconnects	EEPROM	EEPROM	Flash + SRAM
Static current	9 mA–450 mA	9 mA–150 mA	2 mA–50 mA
Technology	0.22 u CMOS EEPROM 4-layer metal (7000 B)	0.3 u, 4-layer metal	0.18 u, 6-layer metal

Table. 14.3. CPLDs dari Xilinx.

Family	*XC9500 (XV, XL, –)*	*CoolRunner XPLA3*	*CoolRunner II*
Macrocells	36–288	32–512	32–512
System gates	800–6,400	750–12,000	750–12,000
I/O pins	34–192	36–260	33–270
Max. internal clock frequency	222 MHz	213 MHz	385 MHz
Building block	GAL 54V18 (XV, XL) GAL 36V18 (–)	PLA block	PLA block
Supply voltage	2.5 V (XV), 3.3 V (XL), 5 V	3.3 V	1.8 V
Interconnects	Flash	EEPROM	
Technology	0.35 u CMOS	0.35 u CMOS	0.18 u CMOS
Static current	11–500 mA	<0.1 mA	22 uA–1 mA

FPGA sangat canggih, dibuat dengan teknologi terkini 0.09 m CMOS dengan 9 lapisan keping dan lebih dari 1,000 I/O pins sudah tersedia. Beberapa perusahaan pembuat FPGA seperti Xilinx, Actel, Altera, QuickLogic, Atmel, dll. Contoh dari dua perusahaan (Xilinx dan Actel dapat dilihat pada tabel 14.4 dan 14.5 yang berisikan ribuan *flip-flops* dan jutaan gerbang.

Tabel 14.4. FPGA dari Actel

Family	*Accelerator*	*ProASIC*	*MX*	*SX*	*eX*
Logic modules	2,016–32,256	5,376–56,320	295–2,438	768–6,036	192–768
System gates	125 k–2 M	75 k–1 M	3 k–54 k	12 k–108 k	3 k–12 k
I/O pins	168–684	204–712	57–202	130–360	84–132
Flip-flops	1,344–21,504	5,376–26,880	147–1,822	512–4,024	128–512
Max. internal frequency	500 MHz	250 MHz	250 MHz	350 MHz	350 MHz
Supply voltage	1.5 V	2.5 V, 3.3 V	3.3 V, 5 V	2.5 V, 3.3 V, 5 V	2.5 V, 3.3 V, 5 V
Interconnects	Antifuse	Flash	Antifuse	Antifuse	Antifuse
Technology	0.15 u 7-layer metal CMOS	0.22 u 4-layer metal CMOS	0.45 um 3-layer metal CMOS	0.22 u CMOS	0.22 u CMOS
SRAM bits	29 k–339 k	14 k–198 k	2.56 k	n.a.	n.a.

Tabel 14.5. FPGA dari Xilinx.

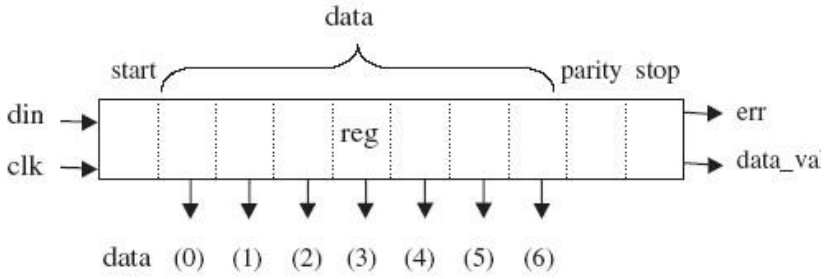
Family	*Virtex II Pro (X)*	*Virtex II*	*Virtex E*	*Virtex*	*Spartan 3*	*Spartan IIE*	*Spartan II*
Logic blocks (CLBs)	352–11,024	64–11,648	384–16,224	384–6,144	192–8,320	384–3,456	96–1,176
Logic cells	3,168–125,136	576–104,882	1,728–73,008	1,728–27,648	1,728–74,880	1,728–15,552	432–5,292
System gates		40 k–8 M	72 k–4 M	58 k–1.1 M	50 k–5 M	23 k–600 k	15 k–200 k
I/O pins	204–1,200	88–1108	176–804	180–512	124–784	182–514	86–284
Flip-flops	2,816–88,192	512–93,184	1,392–64,896	1,392–24,576	1,536–66,560	1,536–13,824	384–4,704
Max. internal frequency	547 MHz	420 MHz	240 MHz	200 MHz	326 MHz	200 MHz	200 MHz
Supply voltage	1.5 V	1.5 V	1.8 V	2.5 V	1.2 V	1.8 V	2.5 V
Interconnects	SRAM	SRAM	SRAM	SRAM	SRAM	SRAM	SRAM
Technology	0.13 u 9-layer copper CMOS	0.15 u 8-layer metal CMOS	0.18 u 6-layer metal CMOS	0.22 u 5-layer metal CMOS	0.09 u 8-layer metal CMOS		
SRAM bits (Block RAM)	216 k–8 M	72 k–3 M	64 k–832 k	32 k–128 k	72 k–1.8 M	32 k–288 k	16 k–56 k

Seluruh FPGA dari Xilinx menggunakan SRAM untuk menyimpan interkoneksi, sehingga mereka dapat diprogram ulang, akan tetapi volatile (sehingga memerlukan ROM eksternal). Sebaliknya, FPGA dari Actel adalah non-volatile (menggunakan antifuse), tetapi tidak dapat diprogram ulang (kecuali dalam satu keluarga, yakni Flash Memory). Masing-masing memiliki keunggulan dan kekurangan, aplikasi yang sesungguhnya akan mengacu dan memilih arsitektur mana yang paling cocok.

14.2. Percepatan Pengembangan Sistem Tertanam Secara Perangkat Keras

Mengingat kemampuan pemrograman perangkat FPGA, dimungkinkan penggunaan perangkat ini untuk membangun sistem tertanam secara lebih cepat, baik dari sisi pengembangan maupun kinerjanya. Disampaikan berikut pemanfaatan perangkat ini untuk membuat module penerima data serial.

Diagram dari penerima data serial dapat dilihat pada gambar 14.7 berisikan masukan data serial (din) dan keluaran data paralel (data(6:0). Sinyal clock (clk) diperlukan pada masukan. Dua sinyal pengendali dibangkitkan dari rangkaian : err (error) dan data_valid. Larik masukan terdiri dari 10 bits, bit pertama adalah bit awal (start), yang bila tinggi membuat rangkaian memulai penerimaan data. Berikutnya, sebanyak 7 adalah bit aktual data. Bit ke sembilan adalah bit paritas, yang statusnya adalah '0' bila banyaknya bit '1' dalam data genap, atau '1' bila sebaliknya. Ketika penerimaan selesai dilakukan dan tidak terjadi kesalahan, data tersimpan dalam register internal (reg) dipindahkan ke data(6:0) dan keluaran data_valid dibangkitkan.

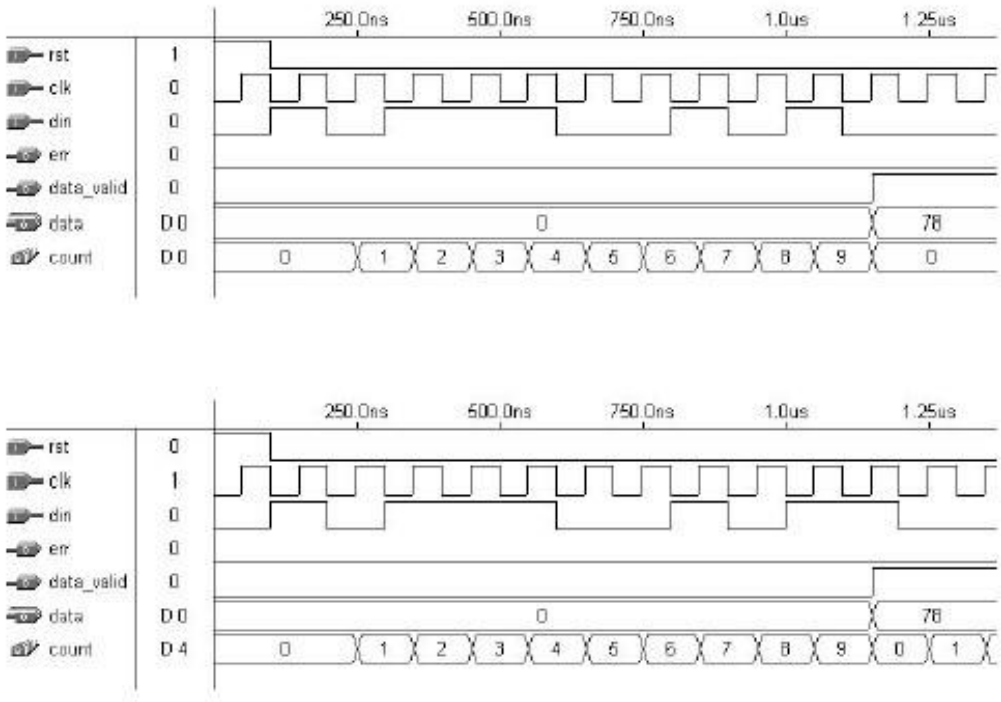


Gambar 14.7. Penerima data serial.

Kode VHDL untuk rangkaian ini disajikan dengan beberapa variabel yang digunakan : *count*, untuk menentukan jumlah bit yang diterima; *reg*, untuk menyimpan data; *temp* untuk menghitung kesalahan. Perhatikan baris ke 37, reg(0)=din bukan reg(0)='0', karena diinginkan slot waktu setelah bit stop secara langsung dianggap sebagai bit awal dari masukan larik berikutnya.

Hasil simulasi disajikan pada gambar 14.8, urutan masukan adalah din={start='1', din = '0111001', parity = '0', stop = '1'}. Tampak pada grafik sebelah atas, tidak terdeteksi adanya kesalahan, karena paritas dan bit akhir adalah benar. Disini, setelah count mencapai 9, data dibuat tersedia, yakni data='0111001', dari data(0) ke data(6), yang berarti merupakan data desimal 78, dan data_valid dibentuk. Perhatikan bahwa keluaran tetap tidak terdefinisi, kecuali masukan baru pada larik diterima. Perbedaan dengan grafik di bagian bawah adalah, bit awal muncul langsung setelah bit akhir. Tampak bahwa variabel count memulai untuk menghitung dan seluruh proses berulang.

Pendekatan lain : dapat digunakan pendekatan FSM (Finite State Machine) baik mesin Moore maupun Meally untuk membangun rangkaian ini, untuk perlu dibuat diagram state yang jelas dari sistem tersebut.



Gambar 14.8. Hasil simulasi penerima data serial.