2008年ベーシックコース12回目

2009.02.21

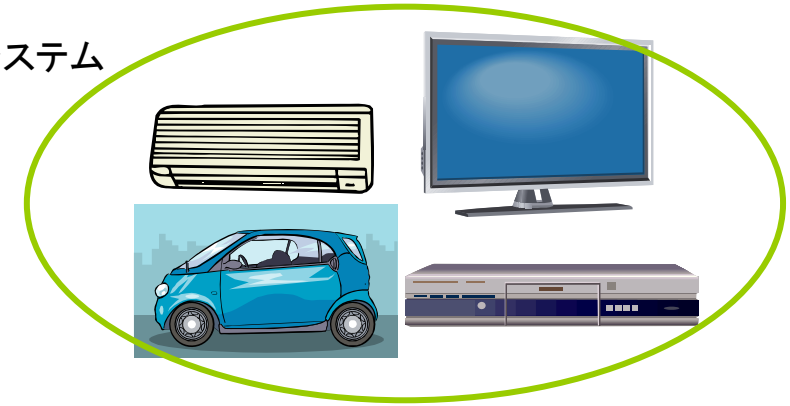
Web用サンプルa版

FPGAの中にARMを作る

奈良高専　土井滋貴

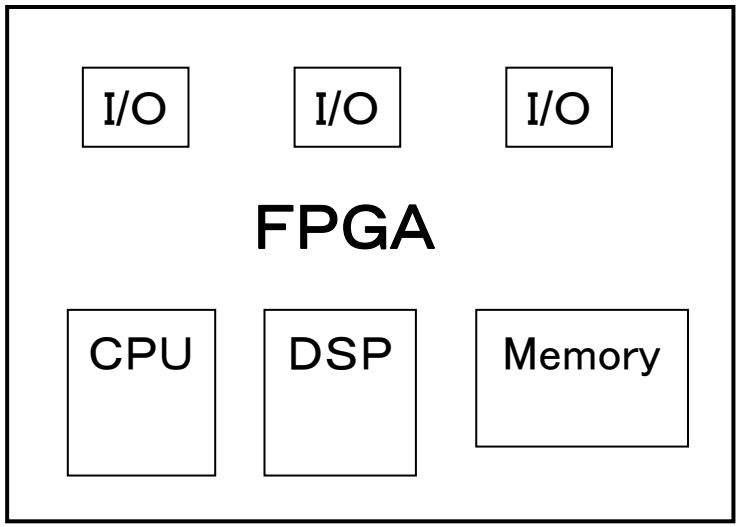
FPGA（Field Programmable Gate Array）と組み込みシステム

組み込みシステム



マイコン搭載回路基板

FPGA等専用LSI



FPGA採用による利点：：

必要な性能のプロセッサーとその周辺回路（ペリフェラル）を柔軟且つ最適に構築できる

具体的には

①手元でプログラミングするだけでよく製造工程なし

②システムのカスタマイズ及び機能向上が迅速且つ容易に実現

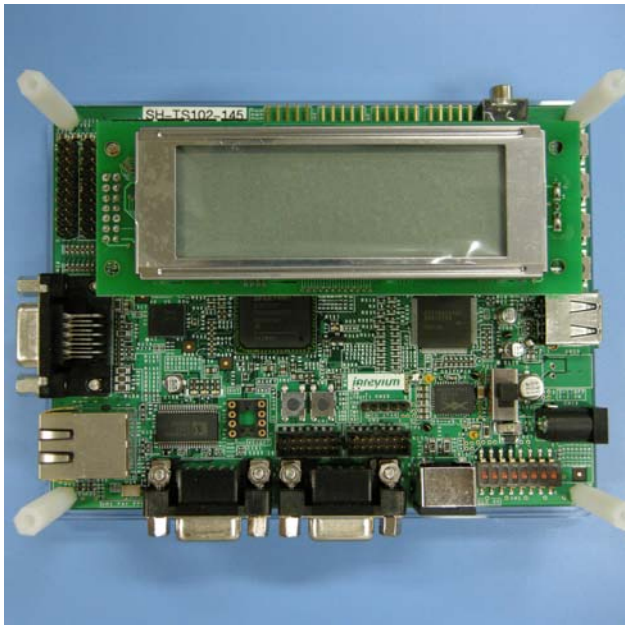
複線的なアプローチ

記述言語

VHDL ⇔ Verilog HDL

チップメーカー

⇔



アルテラ　　ザイリンクス

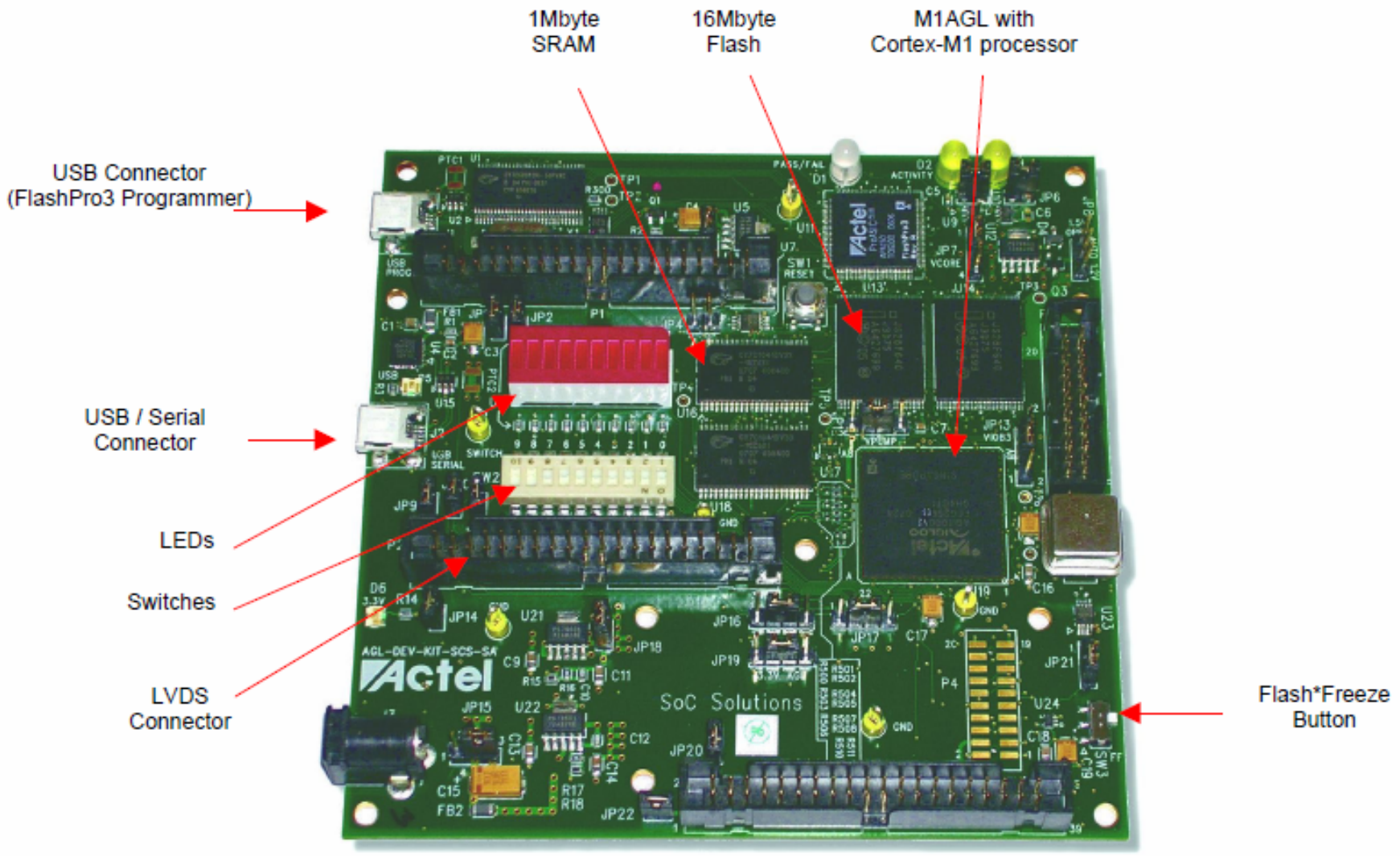
FPGA用CPUコア　Cortex-M1

Cortex-M1

・FPGA実装に向けて開発
・ARMv6-M命令セット･アーキテクチャ
・すべてのThumbコードを実行 - ARM7、ARM9 Thumbサブルーティンを実行
・3段階、32ビットのパイプライン
・メモリーとAHB-Liteインターフェースを分離
・コンフィギュラブルで、ネストされ、ベクターされたインターラプト･コントローラ
・オプションの高速もしくは小型のマルチプライヤー
・Cortex-M3 との上位互換性
・ユーザーによる FPGAへのプログラム書き込み
・Cortex-M1の全I/Oやシグナルにアクセス可能
・アクテルのフラッシュベースM1デバイスに最適化
・ライセンス料やロイヤルティ不要で使用可能
・組み込み、リアルタイムのデバッグ、JTAGインターフェース
・すべての開発ツールがサポート

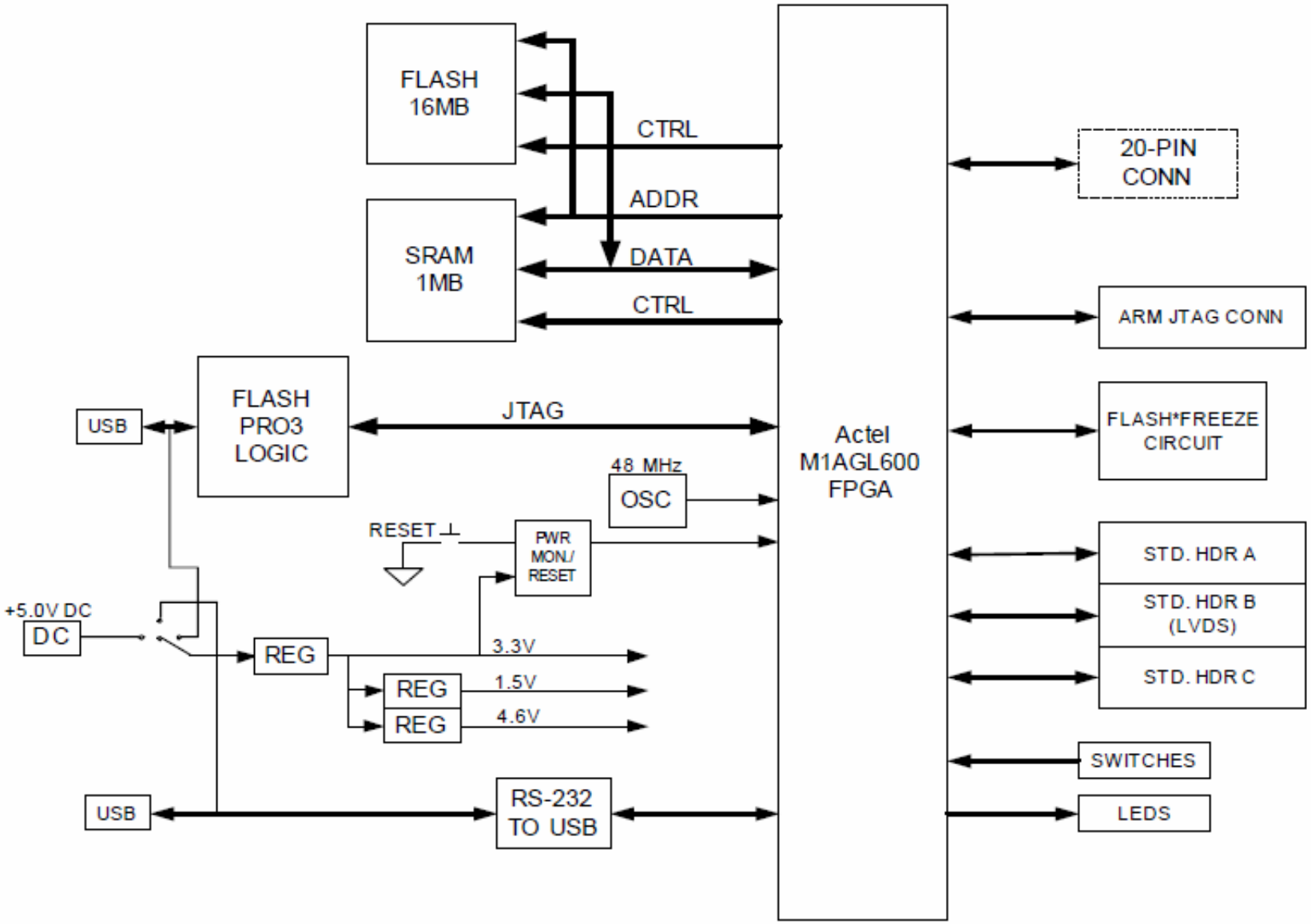
スタータキットの1つ、IGLOO開発キット

IGLOO開発キットに付属するIGLOO開発ボード



M1AGL Development Board

IGLOO開発ボードのブロック図



主要な4つのツール（開発キットに付属するDVDで提供される）

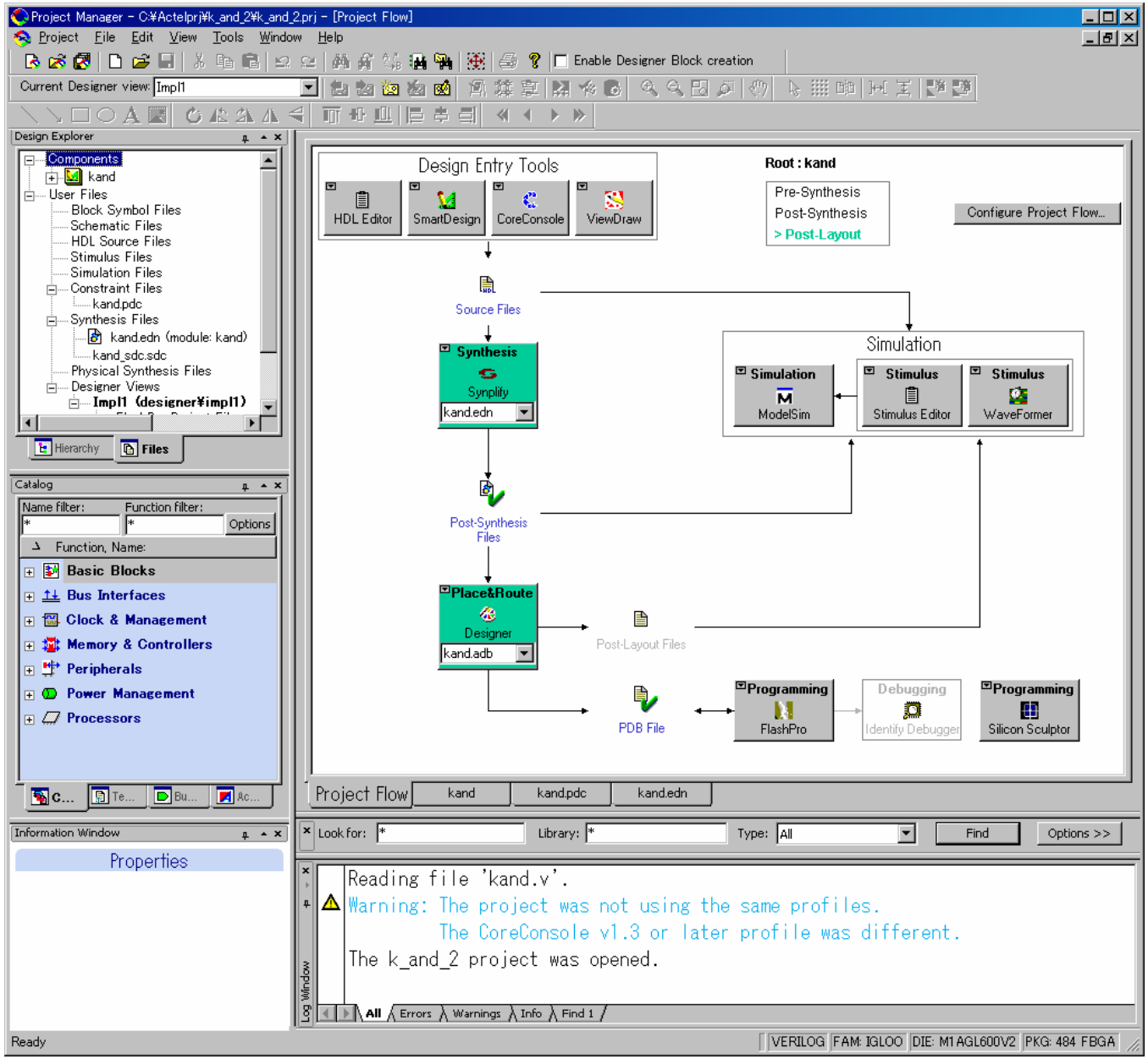
SmartDesign
グラフィックなブロック・システム設計作成ツール
プロセッサ、DirectCores、標準ライブラリ・コア、ユーザIP、およびカスタムHDLを含むFPGAおよびSoC設計の作成
シンセシスが用意されたHDLの自動作成
Libero IDE内での運用

CoreConsole
アクテルのIP設置プラットフォーム（IDP）
FPGAへの組み込みアプリケーション用にIPコアの組立を簡単にするグラフィカル・インターフェースとブロック・スティッチャーを含む
Libero IDEとのシームレス、もしくは独立した運用

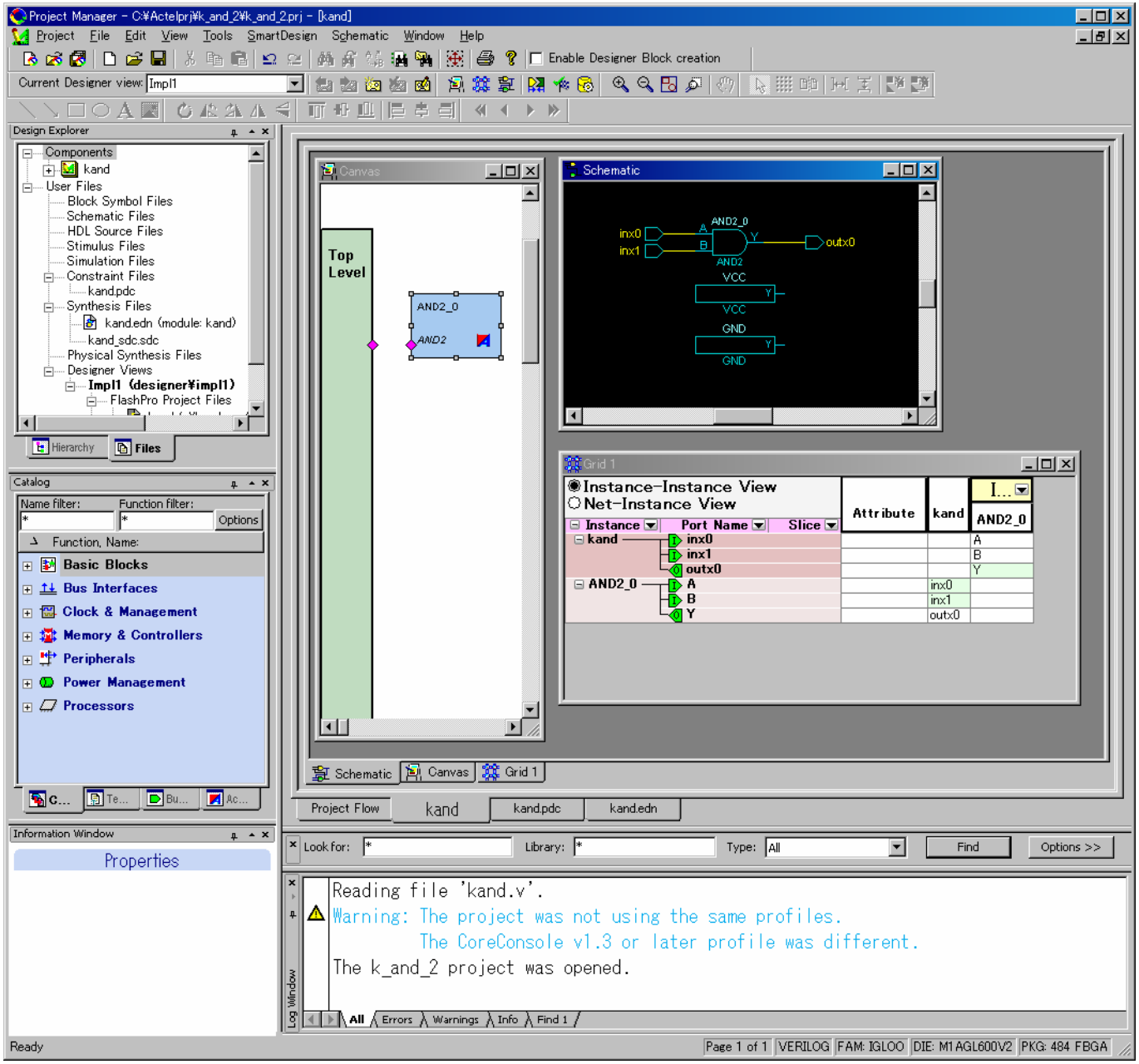
SoftConsole
アクテルのプロセッサ・プログラム開発環境
GNU C/C++ コンパイラ、GDBデバッガ、シミュレータを含む

統合設計環境Libero®
アクテルの包括的なFPGA設計開発ソフトウェア
代表的なEDAベンダーの最新の設計作成、物理的実装、検証ツールを統合

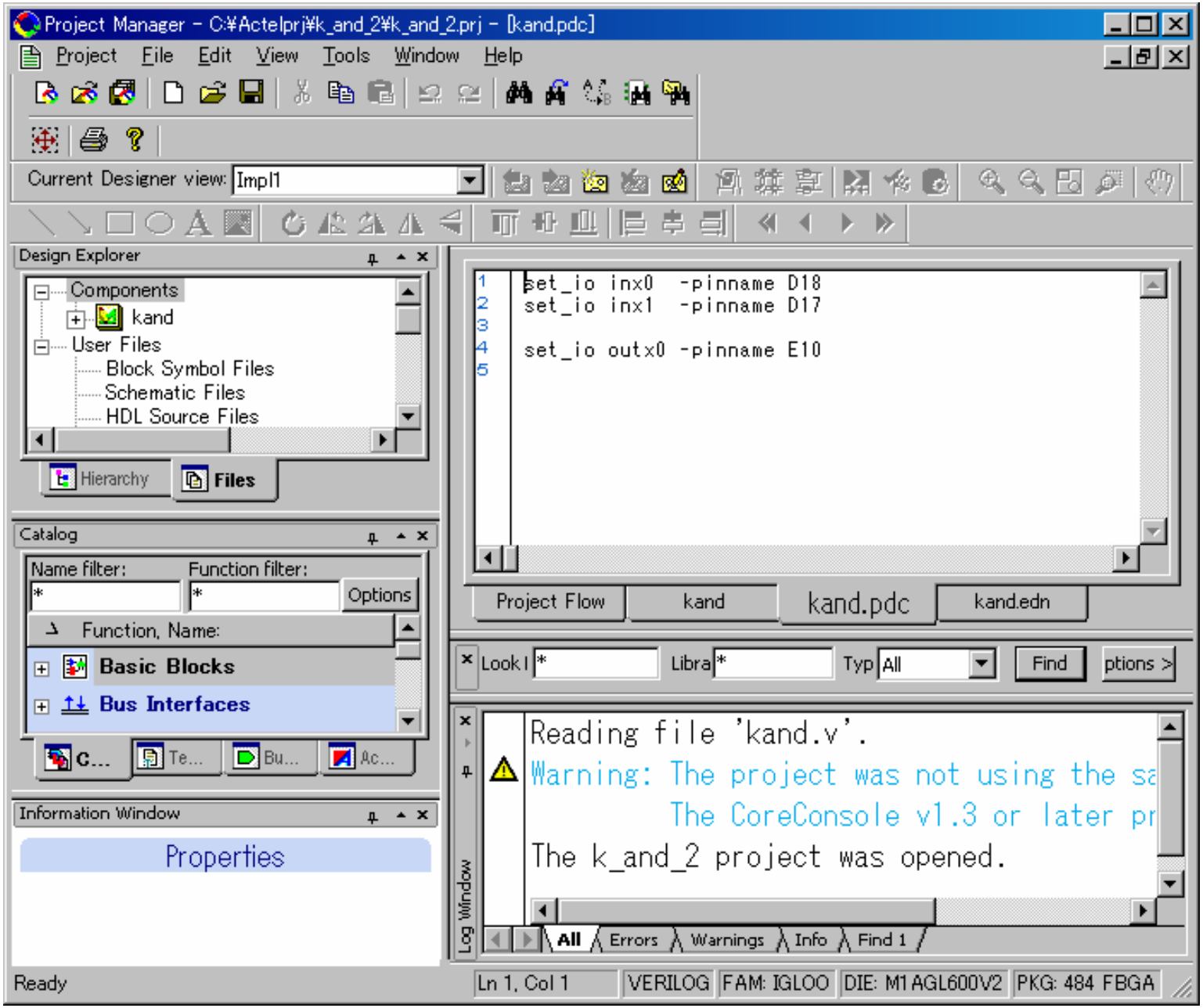
統合環境Libero IDE



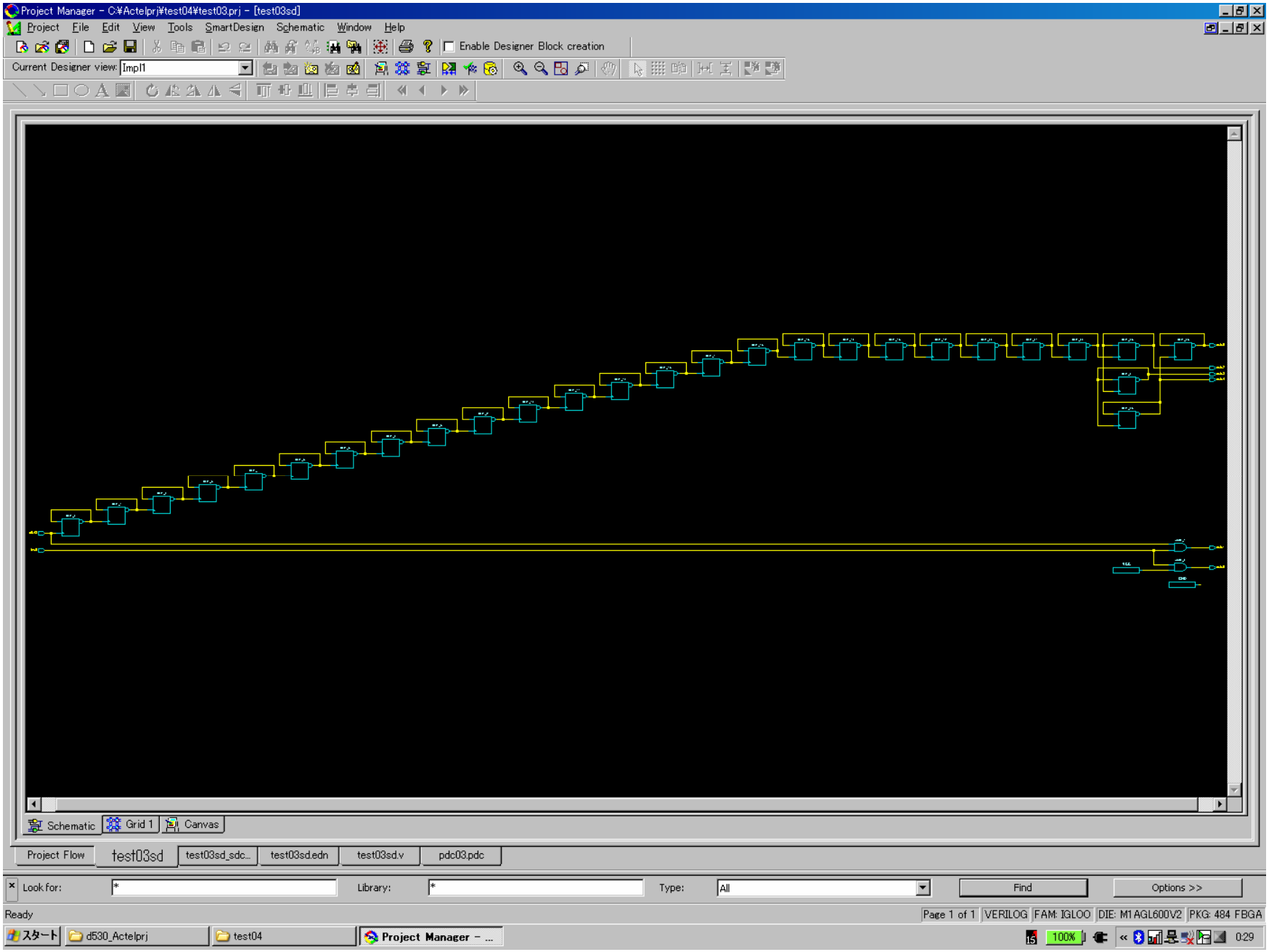
デモ1　回路図入力でAND回路を作る（対象ハードはIGLOO開発ボード）



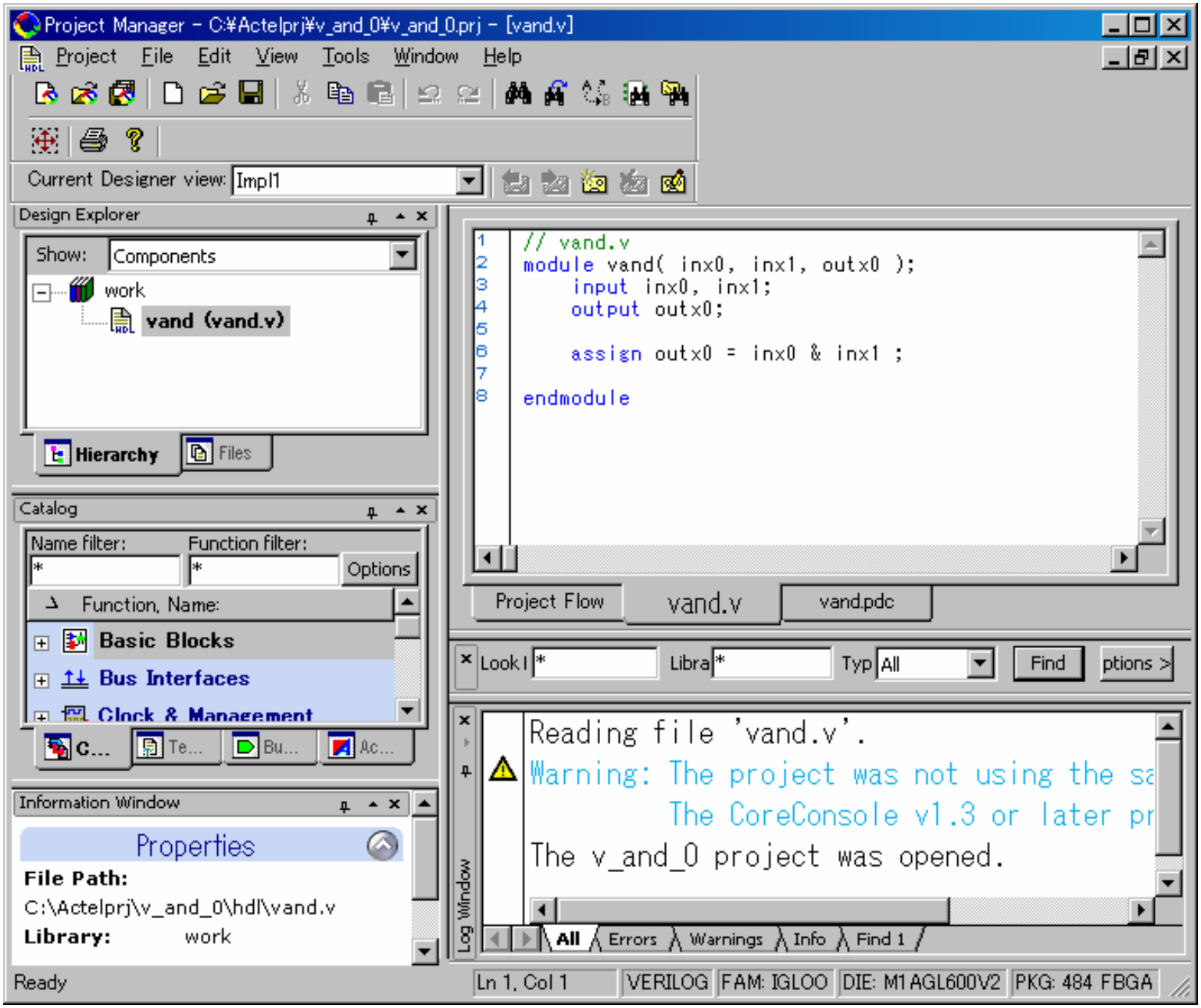
ピンのアサイン（割り振り）はpdcファイルに記述する



デモ2　回路図入力でカウンタを作る（対象ハードはIGLOO開発ボード）

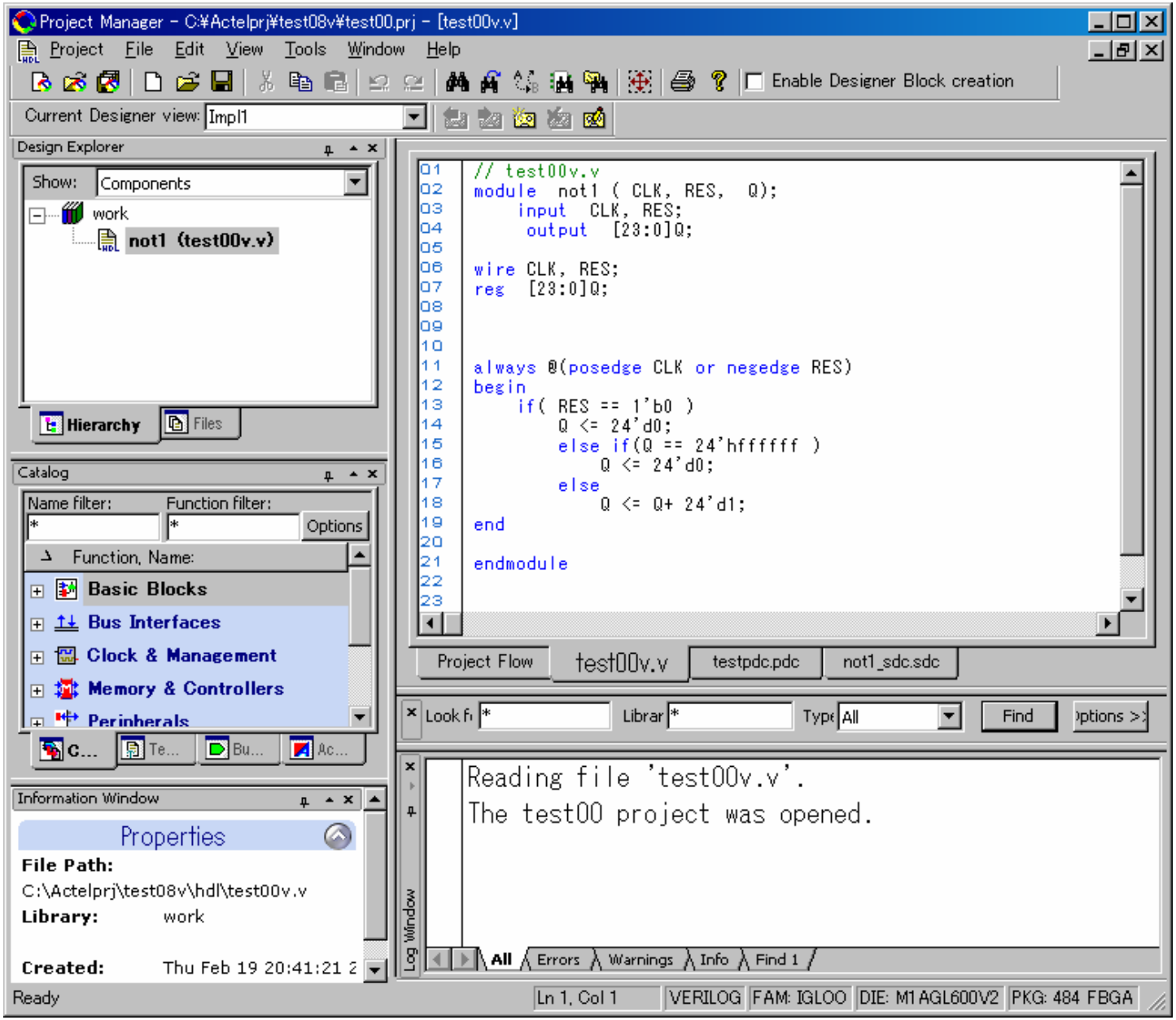


デモ3　VerilogでAND回路を作る（対象ハードはIGLOO開発ボード）



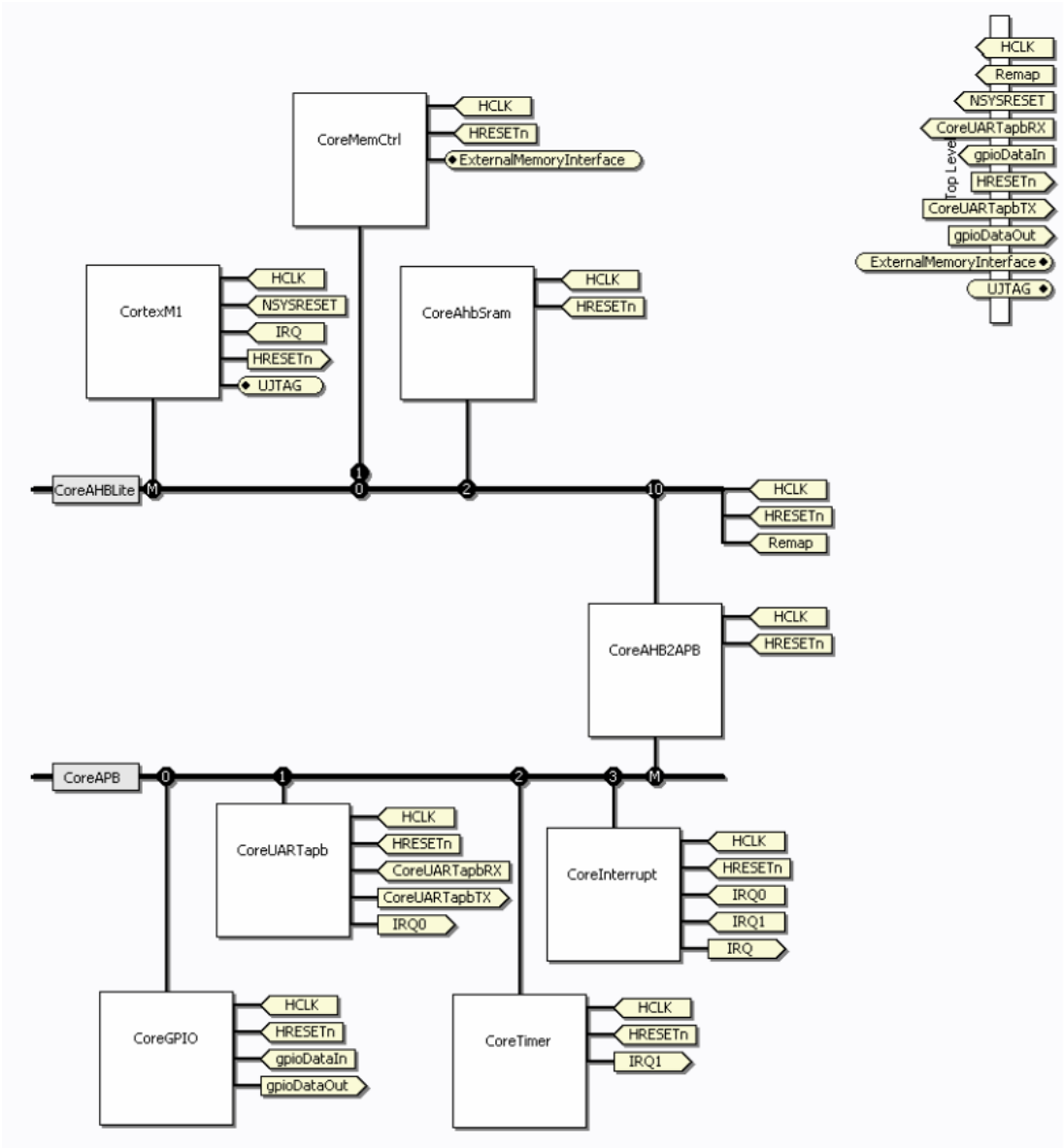
デモ4　Verilogでカウンタを作る（対象ハードはIGLOO開発ボード）

回路図入力に比べて繰り返し等の記述が格段にしやすくなる

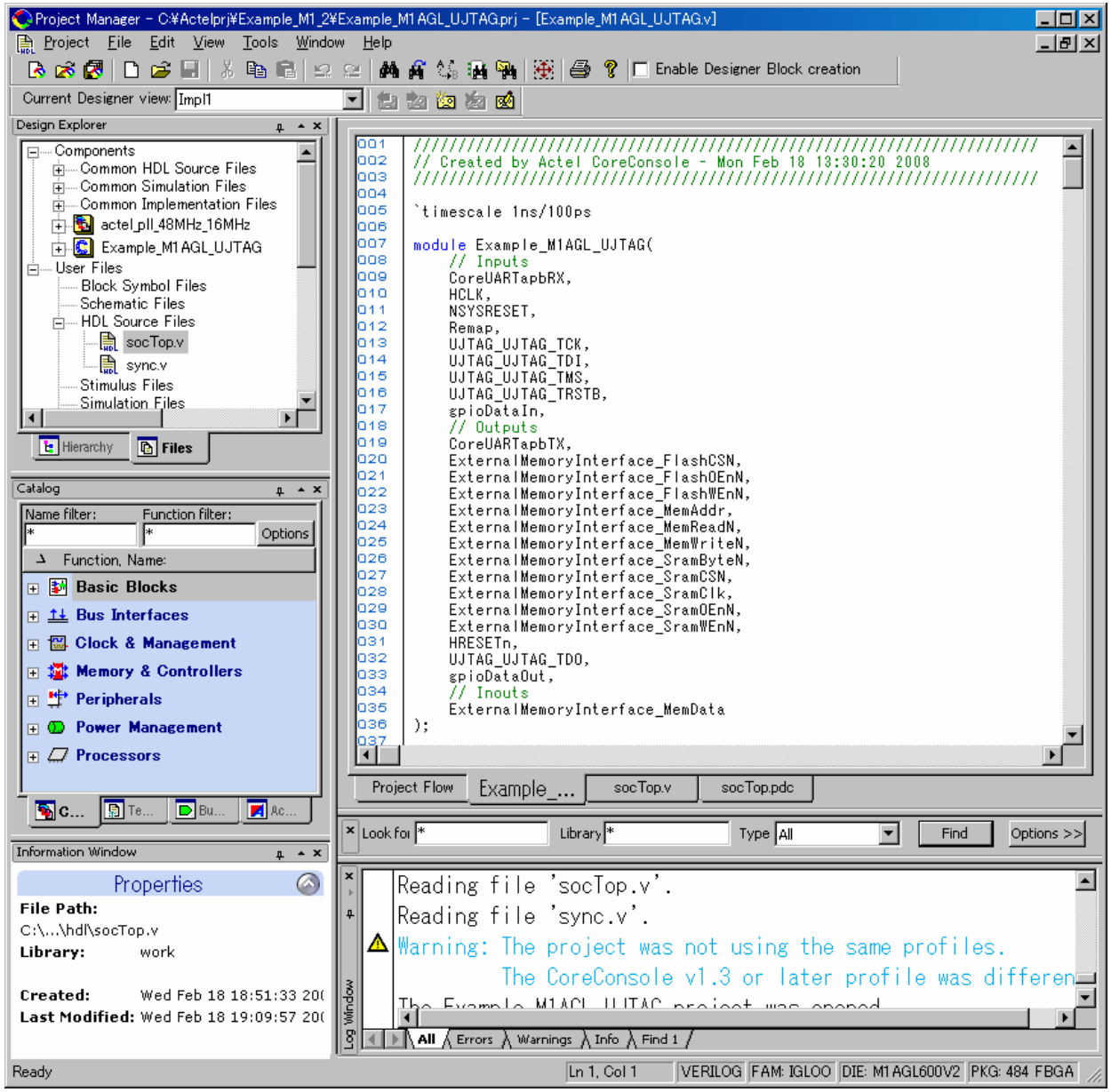


デモ5　M1CPUを作る

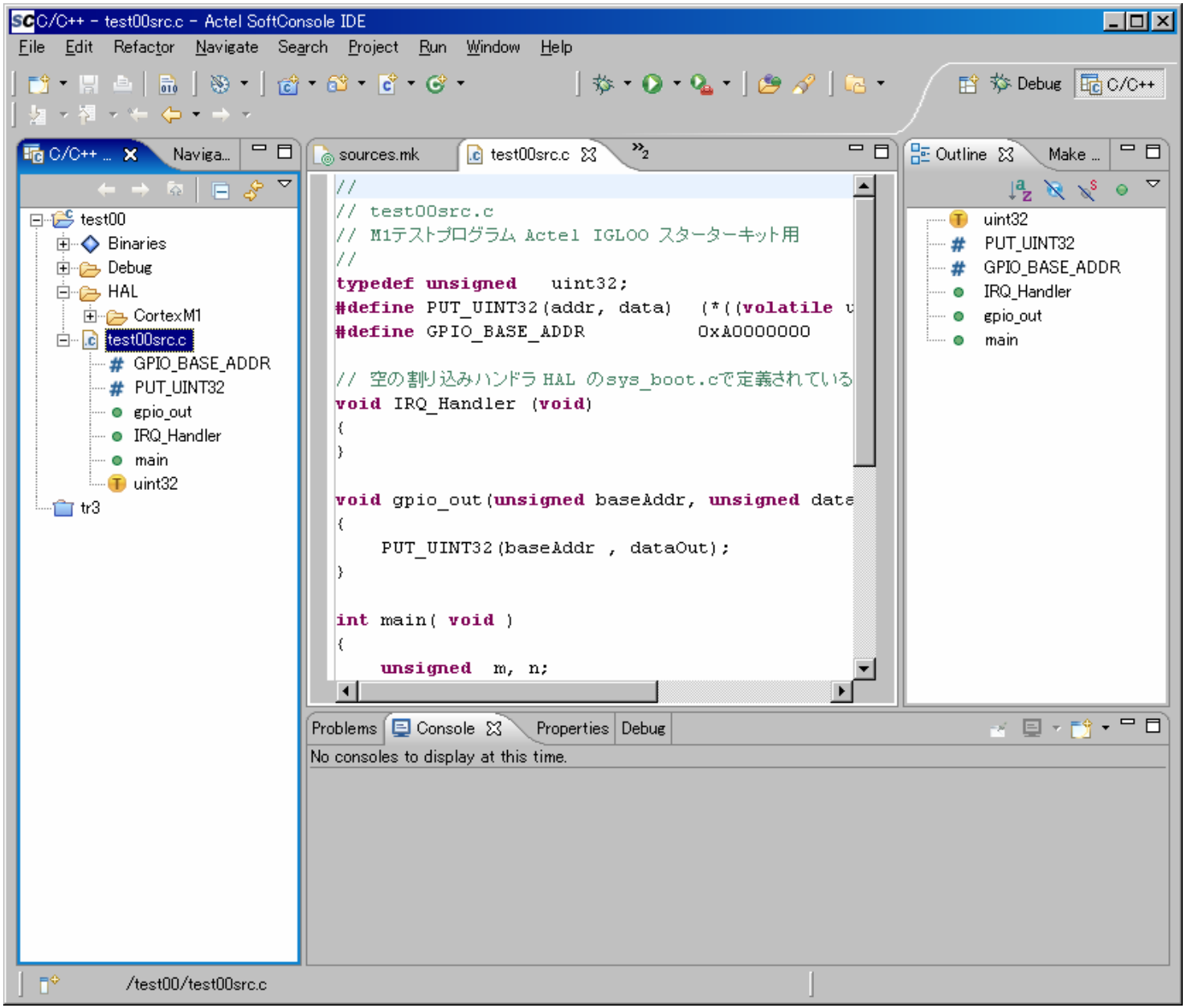
CoreConsoleでCPU周りを設計する。図中のブロック1つ1つが
Corexxxと呼ばれるIP。これらを組み合わせて構築する。



デモ5　M1CPUを作る（対象ハードはIGLOO開発ボード）
CoreConsoleで作成したCPU部分を取り込むTopのHDLを記述する

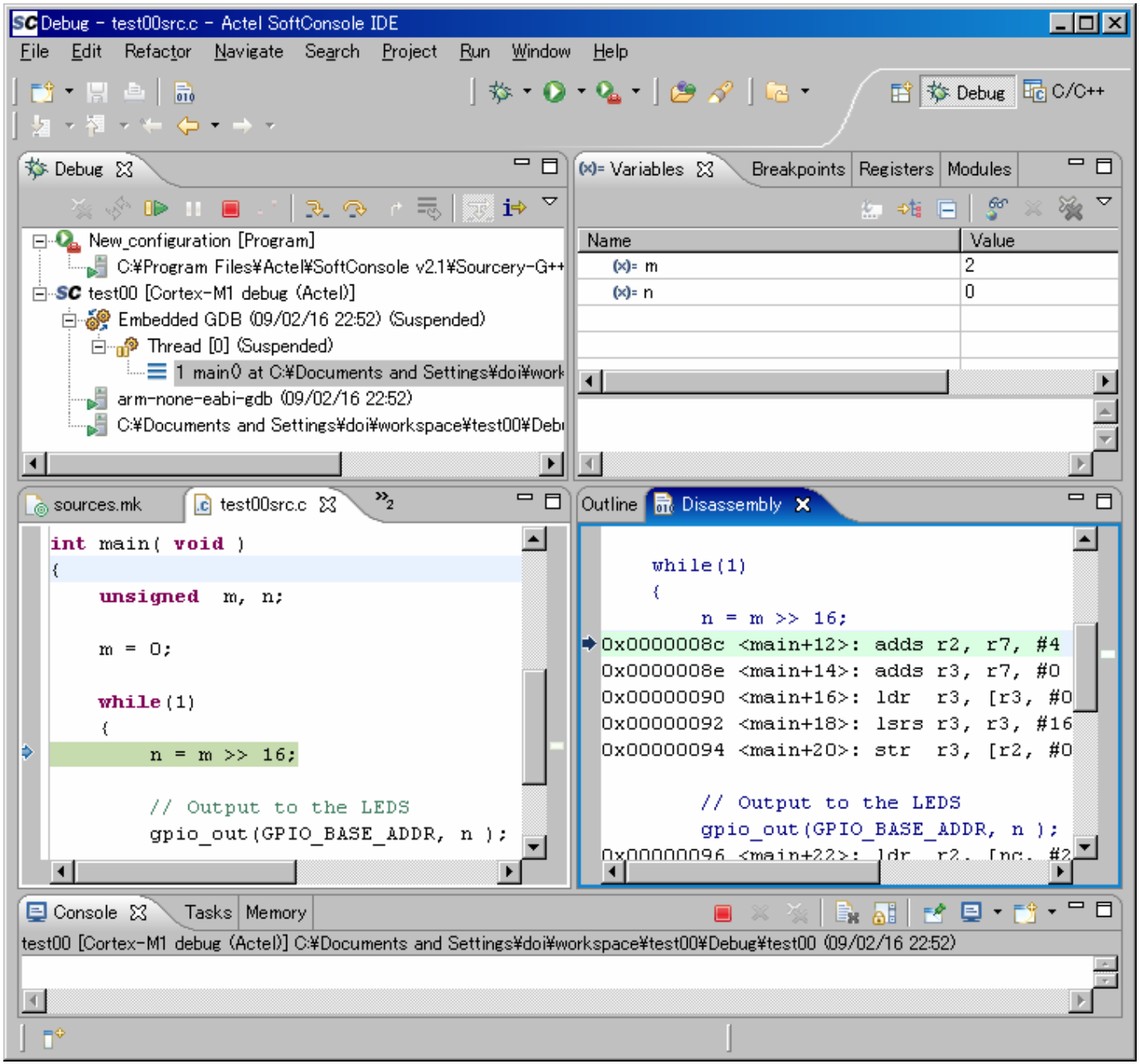


デモ6　M1を使ったカウンタ
構築したCPUで動かすプログラムはSoftConsoleでコーディングする



```
//
// test00src.c
// M1テストプログラム Actel IGL00 スターターキット用
//
typedef unsigned   uint32;
#define PUT_UINT32(addr, data)  (*((volatile uint32 *)
(addr)) = (data))
#define GPIO_BASE_ADDR          0xA0000000

// 空の割り込みハンドラ HAL のsys_boot.cで定義されている
void IRQ_Handler (void)
{
}

void gpio_out(unsigned baseAddr, unsigned dataOut)
{
    PUT_UINT32(baseAddr , dataOut);
}

int main( void )
{
    unsigned  m, n;

     m = 0;

     while(1)
    {
        n = m >> 16;

        // Output to the LEDS
        gpio_out(GPIO_BASE_ADDR, n );

         m++;
    }
}
```

通常の
C言語プログラムと
変わらない

記述はいわゆる
32ビット版記述

JTAGを利用したデバッグ実行の様子