

はじめてのLSI設計 ～HDLでデジタル回路編

秋田純一(金沢大)/MakeLSI:

Contents

- ☑ HDLからのLSI設計の流れ
- ☑ はじめてのLSI設計（HDLでデジタル設計）
 - ☑ HDLの入手
 - ☑ Qflowで半手動設計

HDL→LSIレイアウトへの道のり

HDLで書いた論理回路

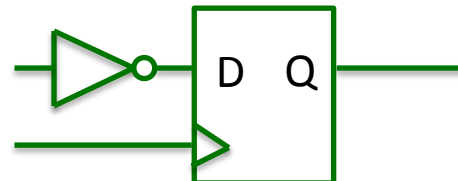
```
always @(posedge ck) begin
  if (rst == 1) cnt <= 0;
  else cnt <= cnt + 1;
end
```

※HDL=Hardware Description Language
(ハードウェア記述言語)

※最近ではC言語から論理合成
(高位合成)も実用化されつつある

論理合成 ツール

ネットリスト(回路図)

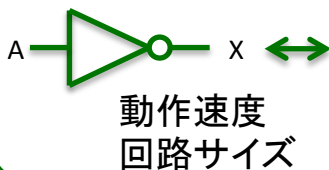


配置配線 ツール

レイアウト図



ライブラリ

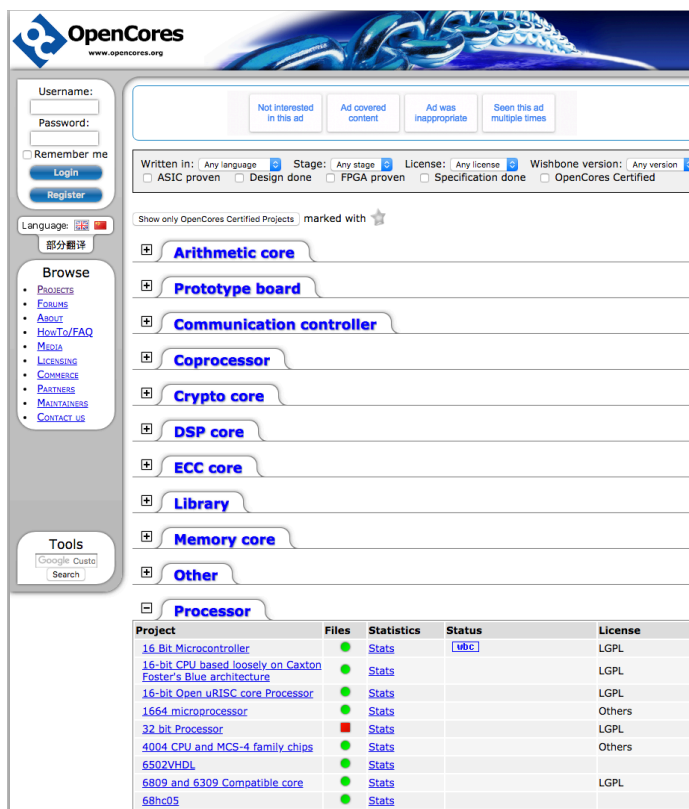


制約条件

- 動作速度
- 回路サイズ など

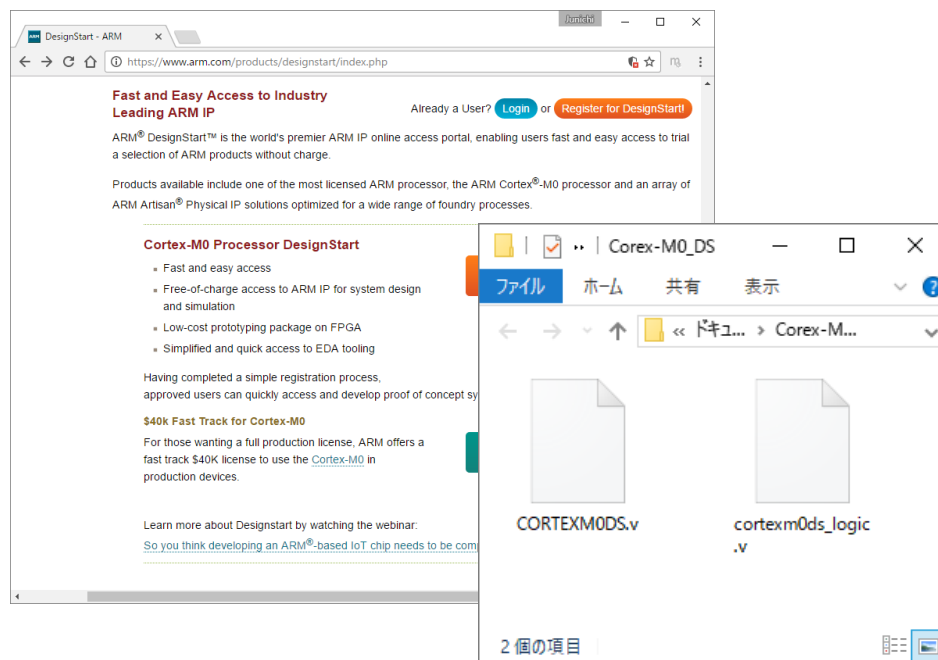
まずはHDLを入手

✓購入(IP)／OpenSourceHW／その他



The OpenCores website displays a variety of IP cores categorized by type, such as Arithmetic core, Prototype board, Communication controller, Coprocessor, Crypto core, DSP core, ECC core, Library, Memory core, and Processor. A table at the bottom lists specific processor projects with their file names, statistics, status, and licenses.

Project	Files	Statistics	Status	License
16 Bit Microcontroller	●	Stats	libc	LGPL
16-bit CPU based loosely on Caxton Foster's Blue architecture	●	Stats		LGPL
16-bit Open uRISC core Processor	●	Stats		LGPL
1664 microprocessor	●	Stats		Others
32 bit Processor	●	Stats		LGPL
4004 CPU and MCS-4 family chips	●	Stats		Others
5502VHDL	●	Stats		
6809 and 6309 Compatible core	●	Stats		LGPL
68hc05	●	Stats		



The ARM DesignStart website provides information about the Cortex-M0 Processor DesignStart, including fast and easy access to industry-leading ARM IP. It mentions that ARM DesignStart is the world's premier ARM IP online access portal, enabling users fast and easy access to trial a selection of ARM products without charge. Products available include one of the most licensed ARM processor, the ARM Cortex-M0 processor and an array of ARM Artisan Physical IP solutions optimized for a wide range of foundry processes.

ARM Cortex-M0 Processor DesignStart

- Fast and easy access
- Free-of-charge access to ARM IP for system design and simulation
- Low-cost prototyping package on FPGA
- Simplified and quick access to EDA tooling

Having completed a simple registration process, approved users can quickly access and develop proof of concept sy...

\$40k Fast Track for Cortex-M0

For those wanting a full production license, ARM offers a fast track \$40K license to use the Cortex-M0 in production devices.

Learn more about Designstart by watching the webinar:
[So you think developing an ARM-based IoT chip needs to be com...](#)

File Explorer (Corex-M0_DS):

- CORTEXM0DS.v
- cortexm0ds_logic.v

2 個の項目

ARM Cortex-M0 DesignStart
(評価用は無償でHDL・量産時に課金)

<https://opencores.org/>

今回のお題:MSP430

✓TexasInstrumentsのマイコンMSP430

✓↑そのOpenSource実装

openMSP430
(VerilogHDL)

MSP430G2302-EP (ACTIVE)

Enhanced Product Mixed Signal Microcontroller



DATASHEET

Mixed Signal Microcontroller, MSP430G2302-EP datasheet (Rev. A)

Download

V6212623 VID

ERRATA

MSP430G2302 Device Erratasheet (Rev. I)

olgirard/openmsp430: The openMSP430 is a synthesizable 16bit microcontroller core written in Verilog.

olgirard / openmsp430

Watch 3 Star 16 Fork 2

Code Issues 1 Pull requests 0 Projects 0 Wiki Insights

The openMSP430 is a synthesizable 16bit microcontroller core written in Verilog.

359 commits 2 branches 0 releases 1 contributor BSD-3-Clause

Branch: master New pull request

Create new file Upload files Find file Clone or download

Commit	Message	Time
olgirard	Fix issue when connecting to UART from Windows10	Latest commit 92c883a on Mar 31
core	Update for proper access to IE1 register with new TI/RH GCC toolchain	3 years ago
doc	Fix documentation typo with 'BCSCTLx/DCOCTL' names and addresses.	9 months ago
fpga	Adjust register mapping of FRAME_SELECT	2 years ago
tools	Fix issue when connecting to UART from Windows10	4 months ago
.gitignore	Improved hardware CLUT with support for all CGA 2bpp palettes	2 years ago
ChangeLog_core.txt	Import Changelog files from SVN release version	3 years ago
ChangeLog_tools.txt	Add new donate picture + update Tools Changelog	2 years ago
LICENSE	Initial commit	4 years ago

<https://github.com/olgirard/openmsp430>

※これをdownloadしておく

今回使うツール: Qflow

☑ Tim Edward氏作の上位設計ツール群

<http://opencircuitdesign.com/qflow/>

☑ 論理合成 + 配置配線 + レイアウト (確認編集)
(配置配線以外は既存のものを流用)

☑ Open Source Software

☑ オープンソースなライブラリつき
(Oklahoma State Univ. の osu035/osu05)

HDLからのLSI設計の一般論

☑半自動(全自動ではない)

☑論理合成、配置配線の各ステップで、
手動設定・試行錯誤が入ることが多い

☑設定パラメータがけっこうだいじ

☑レイアウトの縦横比

☑論理ゲートセルの配置密度(高すぎると配線できない)

☑電源配線をおく密度

initial density	rowsep	未配線	実行時間(place,route)[分]
0.6	0.6	0	8,12
0.6	0.7	0	8,12
0.8	0.2	0	10,9
0.6	0.2	26(Stage2)	9,7
0.8	0.1	156(Stage2)	9,5
1.0	0.1	649(Stage2)	9,7
0.8	0.05	263(Stage2)	11,11
0.6	0.1	43(Stage2)	12,10
0.9	0.2	268(Stage2)	8,8

パラメータによっては配線が終わらないなど

Qflowでの設計: Qflowの準備

☑ Qflowをインストール

☑ <https://scrapbox.io/makelsi/>の
「Qflowのインストール」を参考に

☑ Ubuntu Linux 16.04 (仮想マシンやAWSでもOK)

☑ (インストール済みの共用サーバもあり)

☑ GUI版も入れておくとベター? (今回は使わない)

☑ <https://scrapbox.io/makelsi/>の
「Qflowのインストール」の最後のところを参考に

☑ ただしQflowのバージョンが上がって、
設計パラメータも変えないといけなさそう...

Qflowでの設計：準備

- ✓ Ubuntuにログイン
- ✓ プロジェクトのディレクトリをつくる
 - ✓ 例：~/openmsp430/
- ✓ その中に作業用ディレクトリ(3つ)をつくる
 - ✓ “source”, “layout”, “synthesis”
- ✓ “source”の中に、*.v一式をコピーする
 - ✓ openmsp430-master/core/rtl/verilog/
にある*.vすべて
 - ✓ `$ cp (上記path)/*.v ~/openmsp430/source/`

Qflowでの設計：論理合成

☑プロジェクトディレクトリで以下を実行

```
$ qflow synthesize openMSP430
```

※少し時間がかかる

Qflowでの設計：設定（その1）

✓プロジェクトディレクトリのproject_var.shで
“initial_density” を変更



```
project_var.sh - emacs@mineda.anagix.com
File Edit Options Buffers Tools Sh-Script Help

# set yosys_debug =
# set abc_script =
# set nobuffers =
# set nofanout =
# set fanout_options = "-l 200 -c 30"

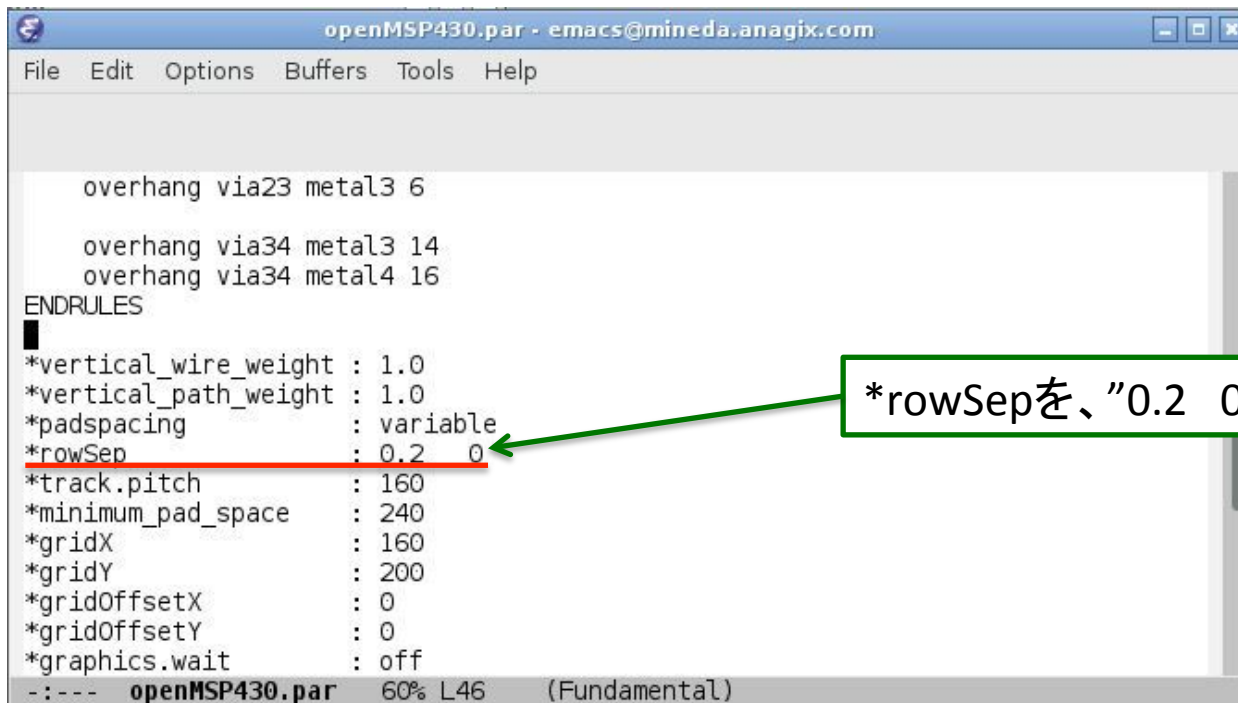
# Placement command options:
# -----
# set initial_density =
# set graywolf_options =
# set addspacers_options = "-stripe 5 150 PG"

# Router command options:
# -----
# set route_show =
```

先頭の#をはずし、0.8に設定
set initial_density = 0.8

Qflowでの設計：設定（その2）

✓ layout/openMSP430.parで
“*rowSep”を変更



```
openMSP430.par - emacs@mineda.anagix.com
File Edit Options Buffers Tools Help

overhang via23 metal3 6
overhang via34 metal3 14
overhang via34 metal4 16
ENDRULES
*vertical_wire_weight : 1.0
*vertical_path_weight : 1.0
*padspacing : variable
*rowSep : 0.2 0
*track.pitch : 160
*minimum_pad_space : 240
*gridX : 160
*gridY : 200
*gridOffsetX : 0
*gridOffsetY : 0
*graphics.wait : off
-:--- openMSP430.par 60% L46 (Fundamental)
```

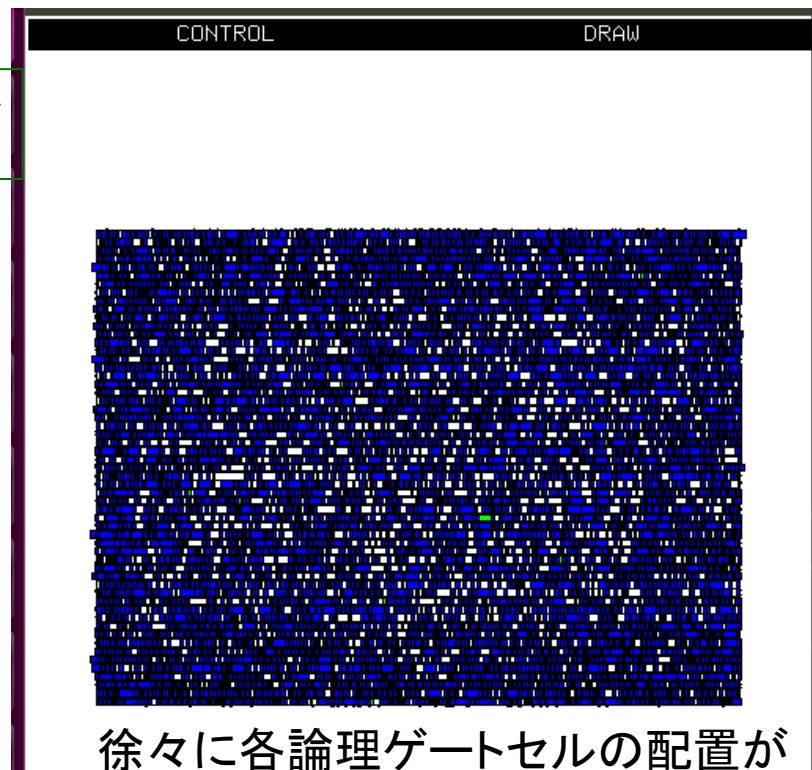
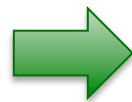
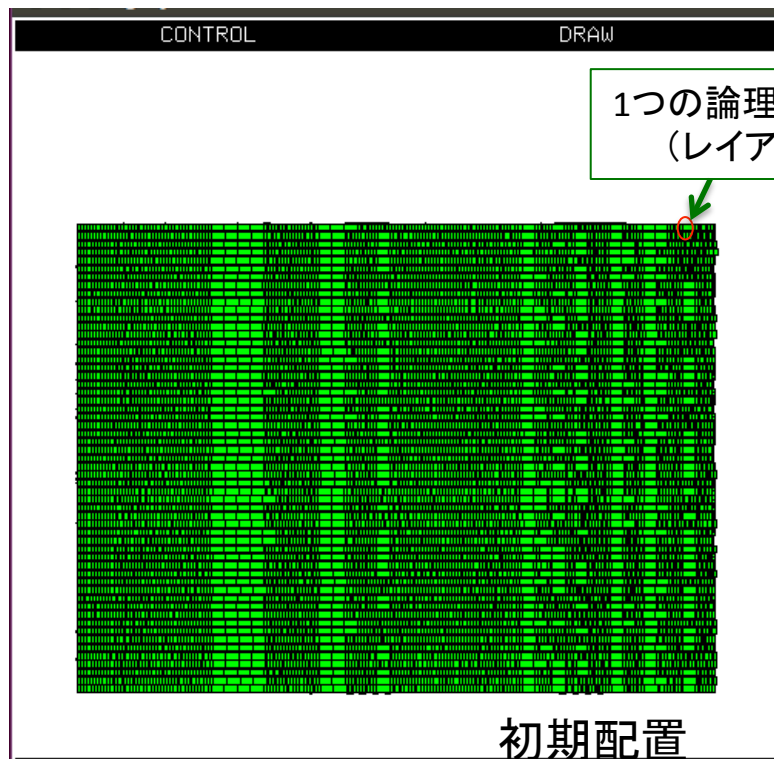
*rowSepを、“0.2 0”に変更

Qflowでの設計：配置

✓プロジェクトディレクトリで以下を実行

```
$ qflow place openMSP430
```

※けっこう時間がかかる

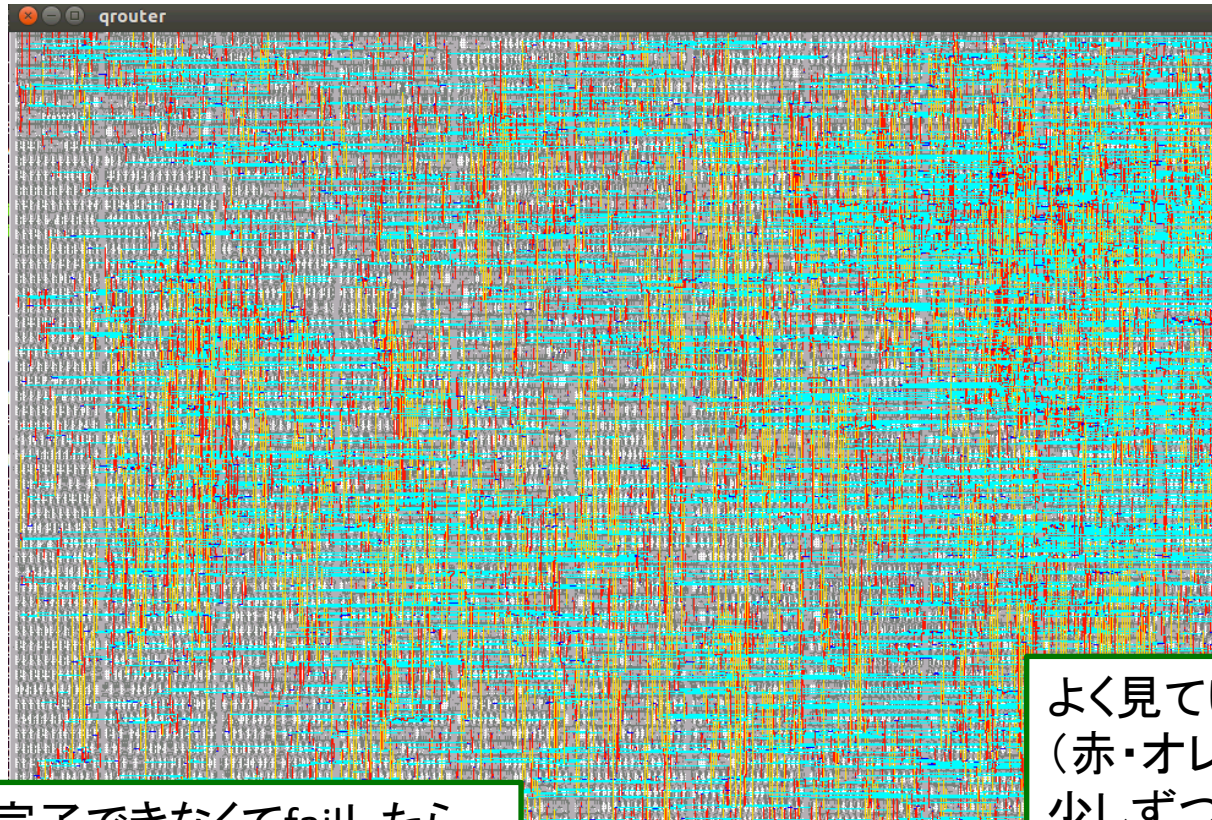


Qflowでの設計：配線

✓プロジェクトディレクトリで以下を実行

```
$ qflow route openMSP430
```

※けっこう時間がかかる



配線が完了できなくてfailしたら、
パラメータを変えて再チャレンジ

よく見ていると、セル間の配線
(赤・オレンジ等の線)が
少しずつ進んでいる

Qflowでの設計:レイアウトの確認

☑プロジェクトディレクトリ/layoutで以下を実行

```
$ magic
```

☑レイアウトツールMagicが起動する

☑※\$ qflow display openMSP430 でもOKそう

☑Magicのコンソールで、以下を設定

※使用する論理ゲートセルのレイアウトと
openMSP430の配置配線の結果を読み込む

```
% lef read /usr/local/share/qflow/  
    tech/osu035/osu035_stdcells.lef (1行で)  
% def read openMSP430.def
```

Qflowでの設計: レイアウトの確認

- ☑ Window→Full View(v)で全体表示
- ☑ 左クリック→右クリックで範囲を選択
- ☑ 全セルを選択後、Cell→Expandで全セル表示

各論理ゲートセル

配線

Expand後の各論理ゲートセルの実体レイアウト

openMSP430の設計結果

- ✓使用ライブラリ: OSU 0.35um
- ✓ゲート数: 9,242
- ✓レイアウト: 1700[um] x 1200[um]
- ✓動作速度: (要調査)

=== openMSP430 ===

Number of wires:	7620
Number of wire bits:	9507
Number of public wires:	7620
Number of public wire bits:	9507
Number of memories:	0
Number of memory bits:	0
Number of processes:	0
Number of cells:	9242

log/synth.log

Total stdcells	:9872
Total cell width	:1.06e+07
Total cell height	:1.97e+07
Total cell area	:2.12e+10
Total core area	:2.12e+10
Average cell height	:2.00e+03

log/place.log