2014년 10월 전자공학회 논문지 제 51권 제 10 호
Journal of The Institute of Electronics and Information Engineers Vol. 51, NO. 10, October 2014 http://dx.doi.org/10.5573/ieie.2014.51.10.096

논문 2014-51-10-14

FPGA를 위한 분석적 배치에서 사전 패킹, 조기 배치 고정 및 밀도 분석 다층화

(Pre-Packing, Early Fixation, and Multi-Layer Density Analysis in Analytic Placement for FPGAs)

김 교 선*

(Kyosun Kim©)

요 약

기존 학계의 FPGA 툴 연구는 단순한 가상 아키텍처 모델 가정에 의존해 왔다. 이러한 제약을 극복하기 위한 첫걸음으로 분석적 배치 및 배치 적법화의 기본 알고리즘들을 상용 FPGA의 아키텍처에 적용하는 실제 상황에서 발생되는 이슈들을 도출하여 대안을 제시한 후 그 효과를 평가하였다. 먼저, 코어 사용률이 낮은 FPGA에서 배치된 셀들의 무게 중심이 칩 중심에서 벗어나는 현상이 발생할 수 있는데 이 변위를 최소화하는 함수를 분석적 배치의 목적 함수에 추가하였다. 또한 배치 밀도 평가의 정확도를 높이기 위해 셀 종류별로 별도의 밀도 행렬을 사용하는 다층 분석, 그리고 자원이 매우 한정된 블록의 조기 고정 방안을 제안하였다. 그밖에, 슬라이스 내에서 두 개의 플립플롭이 제어 핀들을 공유하기 때문에 발생하는 호환성 문제를 개선하기 위한 플립플롭 사전 패킹도 제안하였다. 제안된 기법은 상용 FPGA 아키텍처를 정확하게 모델링하고 수정 개선할 수 있는 K-FPGA 패브릭 평가 툴킷을 근간으로 구현되었으며 12개의 실용 예제에 적용하여 기존 방식에 비해 평균적으로 배선 길이 22%, 슬라이스 사용량 5%를 감축하는 효과를 확인하였다. 본 연구는 신규 FPGA 아키텍처 개발을 위한 최적화 CAD 툴 개발 연구의 기초가 될 것으로 기대한다.

Abstract

Previous academic research on FPGA tools has relied on simple imaginary models for the targeting architecture. As the first step to overcome such restriction, the issues on analytic placement and legalization which are applied to commercial FPGAs have been brought up, and several techniques to remedy them are presented, and evaluated. First of all, the center of gravity of the placed cells may be far displaced from the center of the chip during analytic placement. A function is proposed to be added to the objective function for minimizing this displacement. And then, the density map is expanded into multiple layers to accurately calculate the density distribution for each of the cell types. Early fixation is also proposed for the memory blocks which can be placed at limited sites in small numbers. Since two flip-flops share control pins in a slice, a compatibility constraint is introduced during legalization. Pre-packing compatible flip-flops is proposed as a proactive step. The proposed techniques are implemented on the K-FPGA fabric evaluation framework in which commercial architectures can be precisely modeled, and modified for enhancement, and validated on twelve industrial strength examples. The placement results show that the proposed techniques have reduced the wire length by 22%, and the slice usage by 5% on average. This research is expected to be a development basis of the optimization CAD tools for new as well as the state-of-the-art FPGA architectures.

Keywords : FPGA, Analytic Placement, Multi-Layer Density Analysis, Pre-Packing, Early Fixation

* 정회원, 인천대학교 전자공학과(Department of Electronic Engineering, Incheon National University)
© Corresponding Author(E-mail: kkim@incheon.ac.kr)
※ 이 논문은 인천대학교 2013년도 자체연구비 지원에 의하여 연구되었음.
접수일자: 2014년08월21일, 수정일자: 2014년09월23일, 게재확정: 2014년09월26일

Ⅰ. 서 론

FPGA (Field Programmable Gate Array)에 대한 연구는 IP 검증이나 디지털 시스템의 프로토타이핑 등에 상용 FPGA 칩을 응용하는 분야와 다양한 FPGA 구조에 따라 적합한 논리 합성, 배치, 배선 등의 CAD 툴을 개발하는 분야[1~3]로 양분되어 있다. 상용 FPGA 구조는 개발사의 전유물이고 학계에서 수집되는 정보는 매우 제약적이어서 CAD 툴을 개발하는 연구 대부분은 불행히도 단순한 가상 FPGA 구조를 가정하여 왔다. 최근 다행히도 비교적 많은 정보를 노출하고 있는 Xilinx Spartan/Virtex 계통의 FPGA를 중심으로 툴 개발 환경이 확보되고 있다. USC 그룹의 Torc[4] 및 BYU 그룹의 RapidSmith[5]가 툴킷 형태로 공개된 것이 대표 사례이다. 두 툴킷 모두 Xilinx 사의 XDL 파일 형식을 근간으로 FPGA 구조 데이터베이스를 구축하고 C++/Java 형태의 API (Application Programming Interface)를 제공함으로써 설계 입력, 논리 합성, 패킹, 배치, 배선, 아키텍처 뷰어 등을 플러그-인 할 수 있도록 하고 있다.

국내에서도 2009년부터 한국형 K-FPGA 개발을 국책 과제[6]로 진행하여 왔으며 다양한 기능을 플러그-인 한 툴킷이 개발되었다. 상용 FPGA 구조를 모델링할 뿐만 아니라 이의 변형 또는 새로운 구조를 모델링하여 데이터베이스를 구축할 수 있고 아키텍처 변경에 탄력적으로 대응하기 위해 데이터 구동 형태로 개발된 논리 합성, 배치, 배선, 뷰어, 그밖에 이종 툴과의 인터페이스 등이 이미 플러그-인 되어 있다.

본 논문은 단순한 가상 구조의 가정에 의존해 왔던 기존 학계의 FPGA 연구 한계를 극복하는 첫 걸음으로 상용 FPGA 구조를 정확히 모델링할 수 있는 K-FPGA 툴킷[6] 상에서 실용적 배치를 재구성한다. 특히 초기 배치 (Initial Placement)에 사용되는 분석적 배치 (Analytic Placement)와 배치 적법화 (Legalization)에 관련된 이슈들을 도출하고 이에 대한 해법으로 사전 패킹 (Pre-Packing) 및 조기 배치 고정 (Early Fixation), 그리고 배치 밀도 분산 (Density Distribution)의 정확성 향상을 위한 다층화를 제안한다.

본 논문은 먼저 Xilinx Spartan 구조 및 툴 체인 사용 절차를 Ⅱ장 본론 1절 및 2절에서 소개하고 3절 및 4절에서 분석적 배치 및 적법화 알고리즘을 각각 기술한다. 본론 5절에서는 FPGA 배치의 실질적 이슈 몇 가지를 소개한 후 이슈들을 해결하기 위해 본 논문에서 제안하는 해법들을 간단한 예제를 통해 설명한다. Ⅲ장 실험에서 12개 실용 예제에 대해 적용한 결과를 제시하고 고찰한 후 마지막으로 결론을 맺는다.

Ⅱ. 본 론

1. 상용 FPGA 패브릭 (Fabric) 구조

그림 1에 Xilinx Spartan III[7] 슬라이스 구조를 나타내었다. 먼저 F와 G로 표시된 LUT (Look-Up-Table)가 2개, FFX와 FFY로 표시된 플립플롭이 2개 포함되어 있다. 또한, 입력 수가 많은 광폭 입력 함수 구현을 위한 멀티플렉서 (F5MUX 및 F6MUX), 그리고 산술연산 논리 구현을 위한 캐리 멀티플렉서 (CYMUXF, CYMUXG) 및 전용 게이트들 (XORF, XORG, FAND, GAND)이 있다. 이 밖에 연결 선택을 프로그램 할 수 있는 멀티플렉서, 인버터(INV), 스위치 (USED)들이 연결 경로를 재구성하여 구성 요소들을 선택적으로 사용할 수 있도록 하고 있다. 이러한 슬라이스들이 2차원 배열 구조 형태로 FPGA 칩 상에 수천~수백만 개 분포되며 그것들 사이사이에 배치된 스위치 박스들이 슬라이스들 간의 연결을 결정한다.

최근까지 학계에서는 LUT와 플립플롭만 존재하는 매우 간단한 슬라이스 구조를 가정하여 자동화 툴이 연

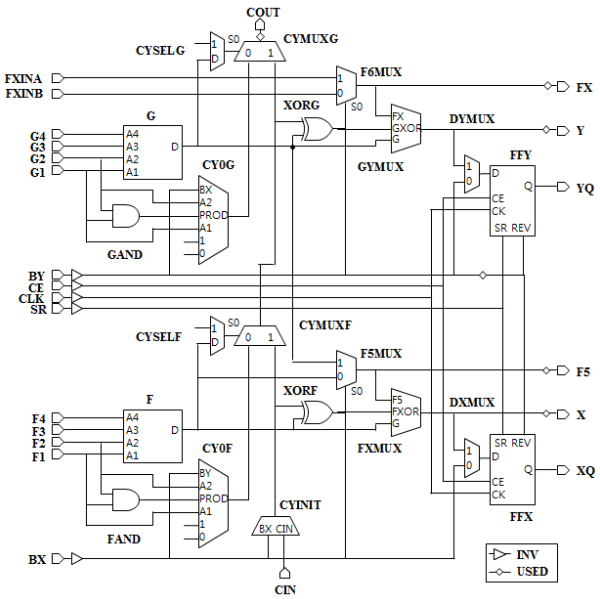


그림 1. Xilinx Spartan III 슬라이스 (SLICEL) 구조
Fig. 1. Slice structure (SLICEL) of the Xilinx Spartan III.

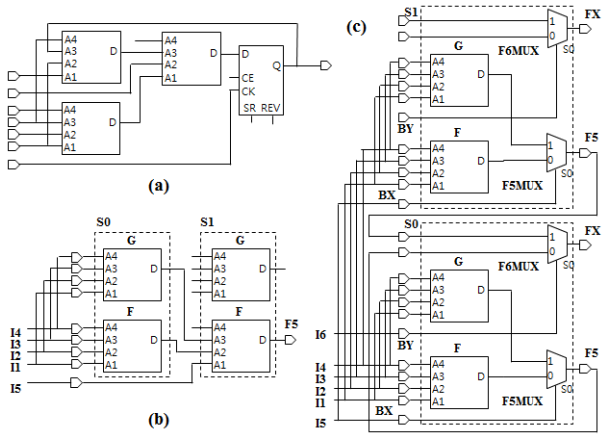


그림 2. 광폭 입력 함수의 FPGA 구현
Fig. 2. FPGA Implementation of a wide fanin function.

구되어 왔다[1~2]. 사실 4-입력 LUT로 입력이 4개인 모든 조합 논리의 구현이 가능하며 조합 논리의 입력 개수가 늘어나도 LUT들을 트리 형태로 연결하면 되므로 LUT만으로 어떤 디지털 회로도 구현할 수 있다. 그림 2(b)에 입력이 5개인 조합 논리 회로를 구현한 LUT 트리를 보이고 있다. 이 트리를 그림 1의 슬라이스에 패킹한다면 2개의 슬라이스 (S0와 S1)가 필요함을 알 수 있다. 만약, Shannon의 확장 정리를 적용하고 그림 1의 슬라이스에 포함된 F5MUX를 사용한다면 그림 2(c)의 S0 슬라이스에 보인 것 같이 2개의 LUT, 즉 하나의 슬라이스만으로 저렴하게 5-입력 조합 논리를 패킹할 수 있다. 더 나아가, S1 슬라이스로 또 하나의 5-입력 조합 논리를 패킹하고 이 둘의 출력을 S0 슬라이스의 F6MUX로 연결하면 6-입력 조합 논리를 구현할 수도 있다. 신호가 하나의 슬라이스에서 다른 슬라이스로 전달될 경우 긴 연결선을 지날 뿐만 아니라 스위치 박스를 하나 이상 거치게 되는데 흔히 이 배선 지연 시간이 LUT의 게이트 지연 시간보다도 훨씬 더 우세하게 된다. 결국 지역 전용 배선과 함께 광폭 입력 멀티플렉서의 사용은 입력이 출력(또는 플립플롭)에 도달할 때까지 지나는 슬라이스 개수 즉 단수를 줄이는 것이 주목적이다.

산술 연산을 수행하는 회로에서는 대개 캐리 체인의 지연 시간이 회로 전체의 성능을 좌우한다. 이 경우에도 슬라이스의 단수를 줄이는 것이 중요하다. 그림 3(a)는 2-비트 리플 캐리 가산기 회로를 나타내었다. 이를 LUT만으로 구현한다면 그림 3(b)와 같이 될 것이며 첫 번째 캐리 출력 C1이 S0 슬라이스에서 S1 슬라이스로

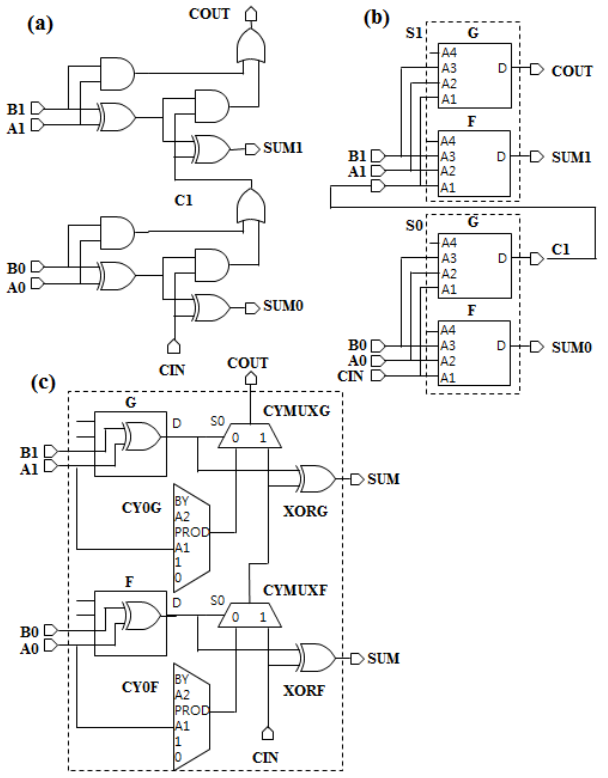


그림 3. 캐리 체인의 FPGA 구현
Fig. 3. FPGA Implementation of a carry chain.

연결되어 2단 회로가 된다. 비트 수가 증가하면 단수는 비트 수만큼 증가할 것이다. 그러나 그림 3(c)와 같이 캐리 멀티플렉서와 전용 XOR 게이트들을 사용하면 한 슬라이스 안에 2-비트 가산기를 구현할 수 있다. 또한, 인접한 슬라이스와 전용 지역 배선을 통해 연결하면 가산기의 비트 수가 증가하더라도 단수 증가를 피할 수 있다.

2. FPGA 설계 툴 체인 사용 절차

그림 4에 FPGA 설계 툴 체인 사용 절차를 나타내고 있다. 먼저 RTL HDL로 기술된 설계가 입력되면 논리 합성기가 논리를 최적화한 후 LUT와 플립플롭 등과 같은 논리 단위 셀 (LUTs & FFs)에 매핑시킨다. 이 때, 산술 연산 전용 게이트, 광폭 입력 함수 멀티플렉서, 블록 메모리, 블록 승산기 등이 넷 리스트에 포함된다. 흔히, 배치는 분석적 배치 및 적법화로 구성되며 최종 패킹 (Final Packer)이 LUT와 플립플롭 등을 슬라이스 내부로 패킹하기 때문에 넷 리스트는 패브릭 단위 셀, 즉, 슬라이스, I/O 버퍼 (IOB), 블록들만의 연결(Slices)로 변환된다. 캐리 체인 및 광폭 입력 함수 구현에 사용

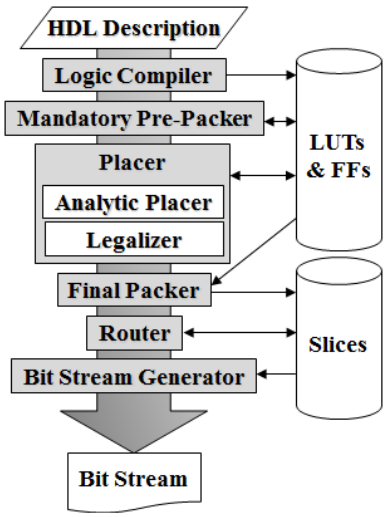


그림 4. FPGA 설계 툴 체인 사용 절차
Fig. 4. FPGA design flow using the tool chain.

되는 논리 단위 셀들은 멀티플렉서 및 전용 게이트들 간의 인접성을 만족시키기 위해 상대적 배치 위치가 미리 결정되며 이를 계속 유지해야 하므로 필수 전처리 과정 (Mandatory Pre-Packer)에서 먼저 그룹으로 패킹하여 그 그룹을 배치에서도 유지한다. 배선은 다양한 길이의 수평, 수직 연결선들 간에 위치한 스위치들의 연결 방향을 선택하여 슬라이스의 핀들을 서로 연결하는 경로를 만들어 내는 과정이다. 배치 배선 후에는 각 스위치들의 상태를 나타내는 비트들을 스트림 형태로 출력한 후 FPGA 칩에 로드할 수 있도록 한다.

3. 분석적 배치 (Analytic Placement)

분석적 배치는 비선형 최적화 기법[8~9]을 이용하며 다음과 같은 목적함수를 사용하는 Conjugate Gradient (CG) Solver[10]가 응용된다.

$$objective = \alpha_l\, length + \alpha_d\, density + \alpha_b\, barrier + \alpha_c\, cogd$$

$$length = \sum_{n\in N}\left(\max_{i\in cells(n)} x_i - \min_{i\in cells(n)} x_i + \max_{i\in cells(n)} y_i - \min_{i\in cells(n)} y_i\right)$$

$$= \sum_{n\in N}\begin{pmatrix} \gamma\log\sum_{i\in cells(n)} e^{\frac{x_i}{\gamma}} + \gamma\log\sum_{i\in cells(n)} e^{-\frac{x_i}{\gamma}} \\ +\gamma\log\sum_{i\in cells(n)} e^{\frac{y_i}{\gamma}} + \gamma\log\sum_{i\in cells(n)} e^{-\frac{y_i}{\gamma}} \end{pmatrix}$$

$$= \sum_{n\in N}\begin{pmatrix} x_{max} + \gamma\log\sum_{i\in cells(n)} e^{\frac{x_i - x_{max}}{\gamma}} + \gamma\log\sum_{i\in cells(n)} e^{-\frac{x_i}{\gamma}} \\ +y_{max} + \gamma\log\sum_{i\in cells(n)} e^{\frac{y_i - y_{max}}{\gamma}} + \gamma\log\sum_{i\in cells(n)} e^{-\frac{y_i}{\gamma}} \end{pmatrix}$$

목적함수는 배선 길이 (length), 배치 밀도 (density), 장벽 (barrier), 그리고 무게 중심 변위 (cogd: Center of Gravity Displacement) 함수로 구성된다. 목적 함수 내 네 가지 함수의 영향을 조정하기 위해 가중치 $\alpha_l, \alpha_d, \alpha_b, \alpha_c$가 사용되었다. 식에서 N은 회로를 구성하는 모든 넷, V는 모든 셀의 집합이며 S는 슬라이스 및 I/O 버퍼 등이 배치될 위치의 집합이다. 변수는 셀 $k\in V$의 X 및 Y 좌표 $x_k, y_k (0 \le k < |V|)$ 로 구성된다. 그리드 집합은 G이며 그리드 $g\in G$의 중심 좌표는 (x_g, y_g)로 표시한다. 배선 길이 (length)는 각 넷의 분포 영역을 외접하는 직사각형 주변 길이의 반 (Half Perimeter)이다. 직사각형의 X 및 Y 구간을 최대 최소 함수로 표현할 수 있으며 이를 지수-합-로그 (Log-Sum-Exponent) 함수로 스무딩(Smoothing)할 수 있다[11]. 지수-합 연산 과정에서 발생할 수 있는 넘침 (Overflow)을 방지하기 위해 $x_{\max}$ 및 $y_{\max}$를 미리 계산한 후 지수 값이 커지지 않도록 빼 주었다가 로그 연산 후에 다시 더해 준다.

$$density = \sum_{g\in G}\left(c_g - d_g\right)^2$$

$$c_g = \sum_{s\in S}\Phi\left(x_s - x_g, y_s - y_g, r\right)$$

$$d_g = \sum_{i\in V}\Phi\left(x_i - x_g, y_i - y_g, r\right)$$

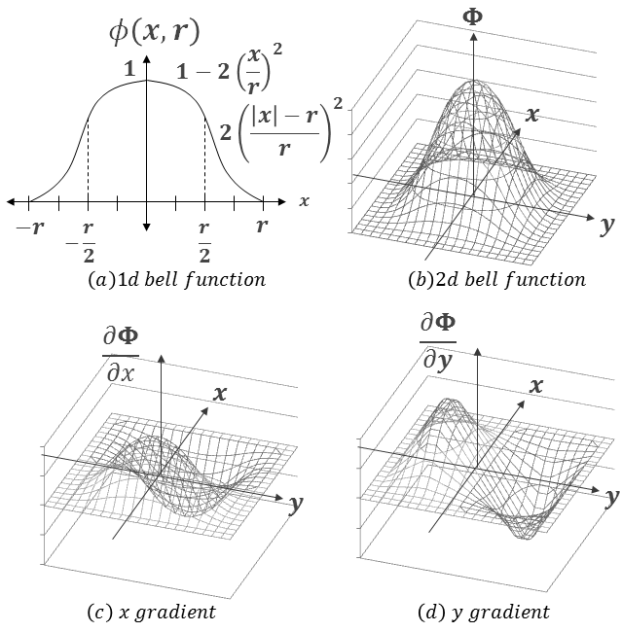
$$\Phi(dx, dy, r) = \phi(dx, r)\cdot\phi(dy, r)$$

$$\phi(d,r) = \begin{cases} 1 - 2\left(\frac{d}{r}\right)^2 & if\ \ |d| < \frac{r}{2} \\ 2\left(\frac{d-r}{r}\right)^2 & if\ \ \frac{r}{2} \le |d| < r \\ 0 & if\ \ |d| \ge r \end{cases}$$

$$\Phi(x,y,r) =$$

$$\begin{cases} 1 - \frac{2}{r^2}x^2 - \frac{2}{r^2}y^2 + \frac{4}{r^4}x^2y^2 & if\ |x|, |y| < \frac{r}{2} \\ \frac{4}{r^4}(|x|-r)^2(|y|-r)^2 & if\ \frac{r}{2} \le |x|, |y| < r \\ \frac{2}{r^2}\left(1 - \frac{2}{r^2}x^2\right)(|y|-r)^2 & if\ |x| < \frac{r}{2}\ and\ \frac{r}{2} \le |y| < r \\ \frac{2}{r^2}(|x|-r)^2\left(1 - \frac{2}{r^2}y^2\right) & if\ \frac{r}{2} \le |x| < r\ and\ |y| < \frac{r}{2} \\ 0 & if\ |x| \ge r\ or\ |y| \ge r \end{cases}$$

배치 밀도 (density)는 종 모양 (Bell-Shape)으로 스무딩하여 각 그리드에 누적시키고 (d_g) 각 그리드의 허



(a)1d bell function (b)2d bell function

(c) x gradient (d) y gradient

그림 5. 종 모양 함수
Fig. 5. Bell-shape function.

용 밀도 (c_g)와의 차를 제곱하여 모든 그리드에 대해 합산한다. 1차원 및 2차원 종 모양 함수는 그림 5 (a) 및 (b)와 같으며 그림 5 (c) 및 (d)에 X 및 Y에 대한 편미분 함수 그래프를 나타내었다.

$$\begin{aligned} &barrier \\ &= \sum_{i\in V}\big(penalty(x_i-l,\alpha)+penalty(r-x_i,\alpha) \\ &\quad +penalty(y_i-b,\alpha)+penalty(t-y_i,\alpha)\big) \end{aligned}$$

$$penalty(d,\alpha)=\begin{cases}0, & if\ d\geq 0\\ \left(\frac{d}{\alpha}\right)^2, & if\ d<0\end{cases}$$

$$cogd=\left(x_c-\frac{\sum x_i}{|V|}\right)^2+\left(y_c-\frac{\sum y_i}{|V|}\right)^2$$

배치 밀도에 의해 셀들이 패브릭 바깥쪽으로 밀려 나가는 것을 방지하기 위해 장벽 (barrier) 함수가 포함되어 있으나 장벽만으로는 배치 분포의 왜곡을 충분히 해소하지 못하는 문제가 있어 무게 중심 변위 함수 (cogd)를 새롭게 제안하였다. 자세한 내용은 5절에 기술한다. 스무딩을 조절하기 위해 파라미터 γ, r, α가 사용되었다. 비선형 최적화에는 각 변수에 대한 목적 함수의 편미분이 필요하며 이는 다음과 같이 계산된다.

$$\begin{aligned} &\frac{\partial}{\partial x_k}length \\ &=\sum_{n\in nets(k)}\left(\frac{e^{\frac{x_k-x_{max}}{\gamma}}}{\sum_{i\in cells(n)}e^{\frac{x_i-x_{max}}{\gamma}}}-\frac{e^{-\frac{x_k}{\gamma}}}{\sum_{i\in cells(n)}e^{-\frac{x_i}{\gamma}}}\right) \\ &\frac{\partial}{\partial y_k}length \\ &=\sum_{n\in nets(k)}\left(\frac{e^{\frac{y_k-y_{max}}{\gamma}}}{\sum_{i\in cells(n)}e^{\frac{y_i-y_{max}}{\gamma}}}-\frac{e^{-\frac{y_k}{\gamma}}}{\sum_{i\in cells(n)}e^{-\frac{y_i}{\gamma}}}\right) \end{aligned}$$

$$\begin{aligned} &\frac{\partial}{\partial x_k}density \\ &=2\sum_{g\in G}(d_g-c_g)\frac{\partial}{\partial x_k}\Phi(x_k-x_g,y_k-y_g,r) \\ &\frac{\partial}{\partial y_k}density \\ &=2\sum_{g\in G}(d_g-c_g)\frac{\partial}{\partial y_k}\Phi(x_k-x_g,y_k-y_g,r) \end{aligned}$$

$$\frac{\partial}{\partial x}\Phi(x,y,r)=$$

$$\begin{cases} -\frac{4}{r^2}x+\frac{8}{r^4}y^2x & if\ |x|,|y|<\frac{r}{2} \\ sgn(x)\frac{8}{r^4}(|x|-r)(|y|-r)^2 & if\ \frac{r}{2}\leq|x|,|y|<r \\ -\frac{8}{r^4}x(|y|-r)^2 & if\ |x|<\frac{r}{2}\ and\ \frac{r}{2}\leq|y|<r \\ sgn(x)\frac{4}{r^2}(|x|-r)\left(1-\frac{2}{r^2}y^2\right) & if\ \frac{r}{2}\leq|x|<r\ and\ |y|<\frac{r}{2} \\ 0 & if\ |x|\geq r\ or\ |y|\geq r \end{cases}$$

$$\frac{\partial}{\partial y}\Phi(x,y,r)=$$

$$\begin{cases} -\frac{4}{r^2}y+\frac{8}{r^4}x^2y & if\ |x|,|y|<\frac{r}{2} \\ sgn(y)\frac{8}{r^4}(|x|-r)^2(|y|-r) & if\ \frac{r}{2}\leq|x|,|y|<r \\ sgn(y)\frac{4}{r^2}\left(1-\frac{2}{r^2}x^2\right)(|y|-r) & if\ |x|<\frac{r}{2}\ and\ \frac{r}{2}\leq|y|<r \\ -\frac{8}{r^4}(|x|-r)^2y & if\ \frac{r}{2}\leq|x|<r\ and\ |y|<\frac{r}{2} \\ 0 & if\ |x|\geq r\ or\ |y|\geq r \end{cases}$$

배선 길이의 편미분은 해당 셀에 연결된 넷들 (nets(k))에 대해서만 계산되며 배치 밀도의 경우 각 그리드의 허용 밀도(c_g)와 셀 밀도(d_g)를 미리 계산하여 2차원 배열에 저장해 두었다가 각 변수에 대한 편미분 계산 시 이를 반복 사용함으로써 계산량을 최소화하고 있다. 무게 중심 변위에 대한 편미분값은 모든 변수에 대해 동일하며 이는 셀들 전체의 무게 중심을 칩 중앙

(x_c, y_c)으로 옮기기 때문이다.

$$\frac{\partial}{\partial x_k} barrier = \frac{\partial}{\partial (x_k - l)} penalty(x_i - l, \alpha) - \frac{\partial}{\partial (r - x_i)} penalty(r - x_i, \alpha)$$

$$\frac{\partial}{\partial y_k} barrier = \frac{\partial}{\partial (y_k - b)} penalty(y_i - b, \alpha) - \frac{\partial}{\partial (t - y_i)} penalty(t - y_i, \alpha)$$

$$\frac{\partial}{\partial d} penalty(d, \alpha) = \begin{cases} 0, & if\ d \geq 0\ \ or \\ \frac{2}{\alpha^2} d, & if\ d < 0 \end{cases}$$

$$\frac{\partial}{\partial x_k} cogd = \frac{2}{|V|}\left(\frac{\sum x_i}{|V|} - x_c\right)$$

$$\frac{\partial}{\partial y_k} cogd = \frac{2}{|V|}\left(\frac{\sum y_i}{|V|} - y_c\right)$$

필수 사전 패킹에서 캐리체인 및 MUX 트리 등을 구성하는 셀들이 그룹으로 묶이면 앵커 (Anchor)를 기준으로 그룹 내 다른 멤버 셀들의 상대적 위치가 이미 결정된다. 따라서 그룹 배치는 마치 하나의 큰 블록을 배치하는 형식을 취한다. CG Solver는 앵커의 좌표만을 변수에 포함시켜 최적 위치를 결정하지만 멤버들 모두가 배치될 공간이 확보되어야 하고 연결도 되어야 하므로 멤버들도 모두 배선 길이, 밀도, 장벽, 무게 중심 변위를 포함하는 목적 함수계산에서 그 좌표 값들이 참조된다. 따라서 결정된 앵커의 좌표를 기준으로 다른 멤버들의 좌표를 먼저 업데이트 한 후에 이 좌표들을 모두 포함시키는 편미분 계산을 실시해야 한다.

4. 적법화 (Legalization)

FPGA 칩 중앙에는 그림 6과 같이 거대한 2차원 배열 형태의 슬라이스 코어가 위치하고 좌우상하에 각각 1차원 배열 형태로 I/O 버퍼 슬롯이 있다. 블록 메모리와 블록 승산기를 배치할 수 있는 수직 1차원 배열 2개가 슬라이스 코어 중간에 끼어 있으며 클록 멀티플렉서 및 클록 발생기들이 I/O 버퍼 슬롯 배열 중간에 끼워져 있다. 블록은 슬라이스보다 8배 정도 크기 때문에 개수가 많지 않다.

배치 적법화는 분석적 배치 시 결정된 좌표와 적법화 시 선택된 배치 슬롯 좌표와의 변위를 최소화하는 것이 목표이다. 과밀은 이 변위 최소화의 가장 큰 적이다. 분석적 배치에서 과밀이 대부분 해소되지만 과밀이 남아

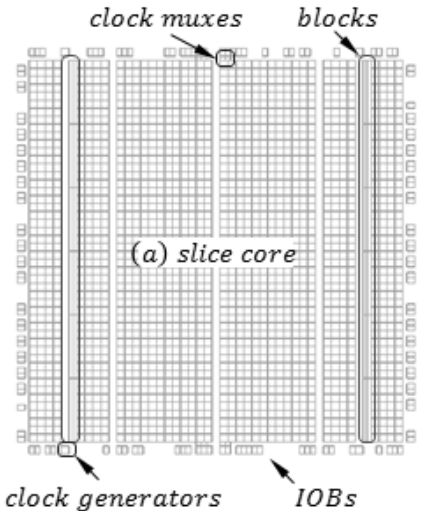


그림 6. 배치 슬롯
Fig. 6. Placement slots.

있다면 적법화 시 해소되어야 하며 이들 중 칩 중앙에 발생한 과밀은 셀들을 최외곽까지 밀어내야 하기 때문에 가장 해소가 어렵다. 따라서 다음과 같은 휴리스틱을 사용한다.

먼저 말단이 충분한 크기의 (예를들어 4x4 이상의) 슬라이스 행렬로 구성되도록 코어를 4분 트리(Quad-Tree) 형태의 빈 (Bin)으로 분할한 후 분석적 배치에 의해 분산된 셀들을 그 위치와 크기에 따라 해당 빈에 삽입한다. 큰 그룹은 4분 트리 상에서 하위 빈에 속하지 못하고 그 그룹을 포용할 수 있는 더 큰 상위 빈에 삽입된다. 각 빈 별로 배치 적법화를 실시하며 미처 적법한 배치를 하지 못한 나머지 셀들은 주변 빈에 넘겨준다. 칩 중앙에서 멀어지는 위치 순서로 빈들이 적법화되며 나머지 셀들은 외곽 빈으로 퍼져 나간다. 각 빈에서도 역시 칩 중앙에서 가까운 슬라이스부터 차례로 최근접 셀을 찾아 채우며 인접 빈에서 밀려난 셀들이 우선 배치된다. 분석적 배치에서 과밀 없이 배치 밀도를 균등히 한다면 대부분의 셀들은 밀려나지 않고 해당 빈에서 적법화 될 수 있다.

5. 이슈 및 해결 방안

가. 자원의 다양성 및 다층 밀도 분석

FPGA에서 조합 논리 소자 (LUT) 및 순차 논리 소자 (FF)는 흔히 같은 비율로 슬라이스에 존재하지만 서로 기능을 바꾸어 사용할 수는 없다. 또한, I/O 버퍼들

과 블록들은 배치 위치가 한정되어 있다. 따라서 밀도 함수에서 공간이 남아 있다 하더라도 다른 종류의 셀 배치에는 무용지물이 될 수 있다. 결국, 밀도 함수 상 소밀 지역이라 하더라도 실제로 특정 종류 셀의 과밀이 존재할 수도 있고 이는 배치 적법화 시 변위가 증가하는 요인이 되며 전체적으로 배선 길이를 증가시킨다.

이밖에 사용 효율 및 성능 향상을 위해 캐리 및 광폭 함수 멀티플렉서, XOR, AND 등의 다양한 전용 게이트들이 슬라이스에 포함되어 있다. 다행히도 이러한 전용 게이트들은 필수 사전 패킹 단계에서 해당 그룹에 소속되기 때문에 앵커의 위치에 따라 배치되고 간단한 조치로 다른 셀들과 겹칠 위험을 방지할 수 있다.

자원의 다양성으로 인한 밀도 함수의 부정확성을 개선하기 위해 다층 밀도 분석 기법을 제안한다. 즉, 그림 7과 같이 LUT, FF, 블록 메모리, I/O 버퍼 셀들에 대해 각각 별도의 밀도 행렬을 사용하여 허용 밀도(c_g) 및 배치 밀도(d_g)를 계산하는 것이다. 따라서 자원별로 허용된 밀도 안에서 셀들이 배치되도록 유도한다.

그림 6과 7에서 확인할 수 있듯이 블록 메모리는 매우 한정된 위치에만 배치할 수 있으며 연결도가 높아 그 좌표를 주어진 영역에서 벗어나지 못하도록 유지하기 어렵다. 따라서 최적화 중간에 이들의 위치를 조기에 고정시키고 다른 셀들이 이들의 위치에 맞추어 배치되도록 하는 방안을 같이 제안한다.

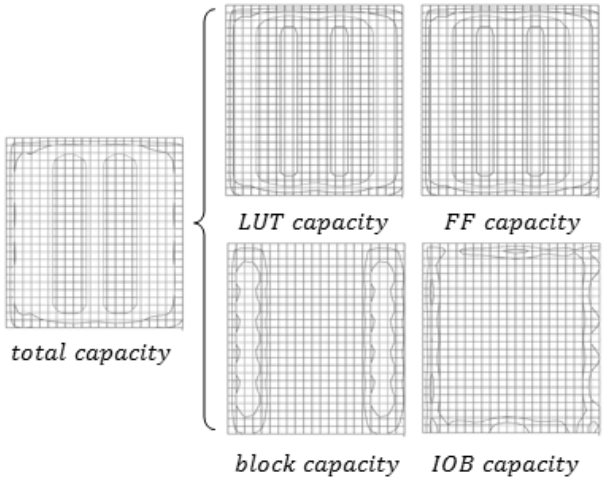


그림 7. 허용 밀도의 다층 밀도 분석
Fig. 7. Multi-layer density analysis for capacity.

나. 자원의 공유 및 호환성

FPGA의 슬라이스는 논리를 효율적으로 구현할 수 있도록 설계되어 있다. 그러나 사실 배선 자원을 효율적으로 운용하기 위한 더 많은 노력이 그 구조 안에 숨겨져 있다. 그것은 소자의 연결에 훨씬 더 많은 자원이 소모되기 때문이다. 예를 들어 그림 2에 나타낸 바와 같이 하나의 슬라이스에는 두 개의 플립플롭이 탑재되며 이들의 클록 (CLK), 클록 허용 (CE), 셋-리셋 (SR), 추가 셋-리셋 (BY) 제어 신호들은 각각 공통의 슬라이스 입력 핀에 연결되어 있다. 따라서 두 플립플롭 간에 ① 이러한 제어 핀에 연결된 넷이 서로 다르든지, ② 하나는 연결되고 다른 하나는 연결되지 않든지, ③ 하나는 동기식이고 다른 하나는 비동기식이면 같은 슬라이스에 공존하지 못한다. 또한, ④ LUT의 출력이 같은 쪽에 있는 플립플롭의 D 입력과 연결되지 않으면 슬라이스의 BY 핀을 사용하여 플립플롭의 D 입력을 연결해야 하므로 추가 셋-리셋을 사용하는 플립플롭은 같은 슬라이스에 배치할 수 없다. 마찬가지로 ⑤ BX/BY 핀은 캐리 MUX의 입력을 공급하기 위해 사용될 수 있는데 이때 같은 슬라이스 내 LUT 출력에 D 입력이 연결되지 않은 플립플롭은 호환성이 없다. 캐리 MUX는 필수 사전 패킹 시 모두 캐리 체인 그룹에 포함되며 이때 체인에 연결된 플립플롭도 포함되기 때문에 규칙 ⑤는 캐리 체인에 다른 플립플롭이 삽입되지 않도록 하는 간단한 조치로 준수될 수 있다. 다른 규칙 ①, ②, ③, ④는 배치 적법화 시 확인하여 호환되지 않는 플립플롭을 배척한다. 그러나 주변에 호환성 있는 플립플롭들이 존재하지 않는다면 하나의 플립플롭만 포함하는 슬라이스들이 많아지게 된다. 이는 슬라이스 이용 효율을 떨어뜨리기 때문에 더 많은 슬라이스가 사용되고 연결 거리가 멀어져 배선 길이도 길어지게 된다.

따라서 본 논문은 호환성이 있는 플립플롭을 사전에 쌍으로 묶어 그룹화하는 방법을 제안한다. 이는 얼핏 연결도가 낮은 셀들을 사전에 묶어 놓음으로써 연결도가 높은 셀들을 근접시키는 배치 최적화 기능을 방해할 것으로 생각할 수 있으나 단 두 개의 셀끼리만 묶기 때문에 그 영향은 매우 미미할 것이며 오히려 사용 슬라이스 수가 줄어들어 배선길이를 감소시킬 수 있다.

다. 무게 중심 변위

분석적 배치는 힘의 균형을 이용하고 있지만 다양한

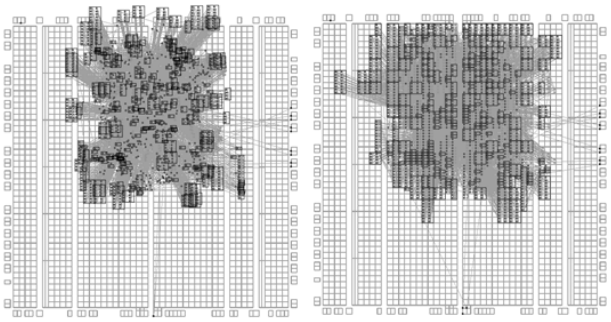


그림 8. 무게 중심 변위 및 그 영향
Fig. 8. Displacement from center of gravity and its effects.

요인에 의해 셀들의 무게 중심이 칩 중앙에서 벗어날 수 있다. 그림 8 (a)와 (b)는 무게 중심이 위로 이동한 분석적 배치 결과 및 이를 적법화한 결과를 각각 보이고 있다. 밀도 함수 및 장벽 함수가 영역 이탈을 억제함에도 불구하고 셀들이 칩 한쪽으로 치우쳐 있으며 칩 경계면에서 과밀이 발생하여 배치가 왜곡되었다.

이러한 무게 중심 변위의 원인은 먼저 ① 밀도 함수에서 c_g가 d_g보다 크면 (소밀의 경우) 그 차를 제곱하지 않고 0으로 계산하기 때문으로 추정할 수 있다. 칩 크기에 비해 슬라이스의 사용량이 적을 경우 만약 이 차를 그대로 제곱해서 계산한다면 셀들이 필요이상으로 분산되어 배선 길이가 증가될 수 있기 때문에 소밀의 경우 0으로 계산한다는 것인데 여기서 힘의 균형이 치우칠 수 있다. 또한 ② 앵커가 움직이면 수직으로 길게 늘어선 그룹 내 셀들이 동시에 이동하기 때문에 기울기가 갑자기 커져 힘의 균형이 손상된다고 추정할 수 있다. 원인에 관계없이 본 논문은 셀들의 무게 중심을 칩 중심으로 향하도록 힘을 가하는 무게 중심 변위 함수를 목적함수에 추가할 것을 제안한다. 해당 함수와 편미분 함수는 이미 3절에 기술되어 있으며 칩 분포가 하나의 둥근 덩어리를 형성할 때 가장 잘 동작한다.

Ⅲ. 실 험

제안된 기법들의 효용성을 확인하기 위해 분석적 배치와 배치 적법화 모듈을 개발하였다. 분석적 배치에는 배선 길이, 밀도, 장벽과 같은 기존 함수들뿐만 아니라 새롭게 제안된 무게 중심 변위 함수도 목적 함수에 포함시킨 후 가중치로 조절할 수 있도록 하였다.

표 1. 12개 예제의 회로 규모 및 소자 사용 프로파일
Table 1. Sizes and device usage profiles of twelve examples.

name	ff	lut	iob	bm	cm	cg
c01bp	4525	6621	111	0	7	0
c02cc	4747	3623	373	11	3	0
c03dr	4864	8920	87	4	2	0
c04em	5575	6109	365	8	1	1
c05fp	5913	5909	199	1	8	0
c06ie	2902	4425	326	10	3	3
c07im	4095	7085	177	4	2	0
c08ip	8286	5566	364	19	5	0
c09mr	9024	10237	370	20	6	3
c10ne	4536	8414	298	0	1	0
c11pc	2902	2801	153	8	2	0
c12sw	10109	7661	400	14	1	0

표 2. 12개 예제의 복잡도
Table 2. Complexity of twelve examples.

name	fan out	clock	distributed RAM			
			s16	s32	sr	d16
c01bp	3.6	7	0	64	0	776
c02cc	2.9	3	5	64	69	0
c03dr	3.5	20	0	0	0	0
c04em	3.5	1	0	0	0	0
c05fp	3.3	29	37	42	0	32
c06ie	3.3	3	0	0	0	0
c07im	3.6	3	294	0	4	744
c08ip	2.8	5	2	8	171	0
c09mr	3.3	16	0	0	0	820
c10ne	3.5	1	0	0	0	0
c11pc	3.2	2	0	0	0	0
c12sw	3.3	1	0	0	0	768

제안된 다층 밀도 분석 기능과 비교하기 위해 기존의 단층 밀도 분석 기능도 구현하여 선택적으로 적용할 수 있도록 하였으며 플립플롭의 제어 핀 공유 제약 조건 준수 기능도 배치 적법화에 탑재하였다. 그밖에 호환성 있는 플립플롭 쌍의 사전 패킹, 블록 메모리의 조기 고정 기능도 선택적으로 적용할 수 있도록 구현되었다. 각 기능의 실행 및 파라미터 조건 변경을 위한 명령들을 구현해 놓고 스크립트로 실행시킬 수 있도록 하였으며 결과를 그래픽으로 확인할 수 있도록 하였다.

Xilinx Spartan III XC3S1000 상용 마스터와 다양한 프로파일의 12가지 예제를 통해 배치 실험을 수행하였다. 배치 결과는 XDL[12]로 출력하여 Xilinx ISE에 입력한 후 배선 및 비트 스트림 발생 기능을 통해 FPGA

표 3. 그리드 해상도, 스무딩, 함수 가중치 파라미터
Table 3. Parameters for grid resolution, smoothing, and function weights.

	slices/grid	γ	r	α_l	α_d	α_b	α_c
1차	4x4	1.5	3	2	1	4	20
2차	2x2	1.5	3.5	1	2	2	20

표 4. 총 배선 길이 비교
Table 4. Comparison of total net lengths.

name	①+②	①+③+④		①+③+④+⑤	
	length	length	%	length	%
c01bp	1019752	735939	72	668969	66
c02cc	1035962	908252	88	757939	73
c03dr	1186833	853682	72	859905	72
c04em	949657	773827	81	744135	78
c05fp	909391	722447	79	631594	69
c06ie	738318	616859	84	588316	80
c07im	820257	626574	76	629427	77
c08ip	1727764	1391764	81	1333449	77
c09mr	2065679	1968307	95	1981547	96
c10ne	1448937	941595	65	989053	68
c11pc	381963	315931	83	344659	90
c12sw	2620497	2589769	99	2257211	86
평균			81		78

칩에 로드할 수 있도록 하였다. 표 1에 12개 예제의 회로 규모 및 소자 사용 프로파일을 요약하였다.

사용된 플립플롭 (ff), LUT (lut), I/O 버퍼 (iob), 블록 메모리 (bm), 클록 MUX (cm), 클록 발생기 (cg) 개수를 나타내고 있다. 표 2는 예제의 복잡도를 요약한다. 넷 당 평균 팬-아웃 수 (fanout), 클록 수 (clock), 사용된 16비트 (s16), 32비트 (s32) 단일 포트 분산 메모리, 쉬프트 레지스터 (sr), 16 비트 2중 포트 분산 메모리 (d16)의 수를 나타내고 있다.

5개 조건들, 즉, ① 무게 중심 변위 함수, ② 단층 밀도 분석, ③ 다층 밀도 분석, ④ 블록 조기 고정, ⑤ 호환성 있는 플립플롭 사전 패킹을 세 가지로 조합하여 분석적 배치 및 배치 적법화를 실시하였다. 조건 ①이 제외되면 배치 및 적법화가 성공하는 조건을 찾기 어렵고 결과도 좋지 않아 조건 ①은 모든 조합에 포함시켰다. 각각 분석적 배치 및 배치 적법화를 교대로 2회 실시하였으며 가중치 등의 파라미터들은 표 3에 요약한 바와 같다. 이는 모든 예제에 동일하게 적용하였다.

세 가진 조건 조합에 대한 배치 결과를 총 배선 길이, 적법화 좌표 변이, 슬라이스 사용량에 대해 표 4, 5, 6에서 각각 비교하였다. 결과 값들은 첫 번째 조합 ① + ②의 결과 값으로 나누어 퍼센트(%)로 표시하였다. 배선 길이는 넷 분포 영역을 외접하는 직사각형 주변 길이의 반 (Half Perimeter)으로 계산하였다.

표 5. 적법화 좌표 변위 비교
Table 5. Comparison of legalization displacement.

name	①+②	①+③+④		①+③+④+⑤	
	displace	displace	%	displace	%
c01bp	709153	460553	65	365867	52
c02cc	561165	424078	76	245601	44
c03dr	742780	381000	51	323009	43
c04em	509709	319291	63	269378	53
c05fp	619765	474922	77	353332	57
c06ie	198167	139817	71	135560	68
c07im	567354	344128	61	302064	53
c08ip	1170620	947332	81	667745	57
c09mr	687720	907240	132	854396	124
c10ne	589960	248901	42	241509	41
c11pc	140863	98595	70	99595	71
c12sw	1243380	1504790	121	1221370	98
평균			76		63

표 6. 슬라이스 사용량 비교
Table 6. Comparison of slice usage.

name	①+②	①+③+④		①+③+④+⑤	
	slices	slices	%	slices	%
c01bp	4104	4084	100	3864	94
c02cc	3298	3317	101	3105	94
c03dr	4833	5000	103	4891	101
c04em	3970	3833	97	3722	94
c05fp	4518	4333	96	4091	91
c06ie	2468	2533	103	2477	100
c07im	4008	4067	101	3956	99
c08ip	5697	5688	100	5167	91
c09mr	6052	5821	96	5837	96
c10ne	4468	4587	103	4523	101
c11pc	1774	1781	100	1798	101
c12sw	6812	6591	97	5425	80
평균			100		95

다층 밀도 분석은 슬라이스 사용량의 변화 없이 평균 19%의 배선 단축, 24%의 적법화 좌표 변위 감소 효과를 보이며 플립플롯 사전 패킹은 추가적인 3%의 배선 단축, 13%의 적법화 좌표 변위 감소, 그리고 5%의 슬라이스 사용량 감축 효과를 올렸다.

비교적 다양한 소자를 포함하고 있으며 평균적 수준의 개선 효과를 보인 c02cc의 각 조건 조합별 배치 결과 레이아웃을 그림 9에 나타내었다. 그림 9 (a)와 (b)

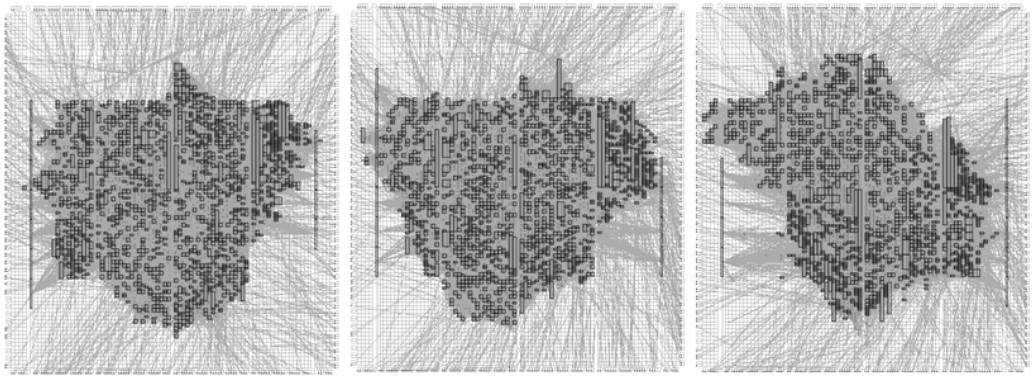


그림 9. c02cc 회로의 3가지 조건 조합에 대한 배치 결과
Fig. 9. Placement results of c02c for three experimental setups.

를 비교하면 다층 밀도 분석은 패턴의 큰 변화를 주지 않으면서 높은 정확도를 통해 좀 더 압축된 결과를 만들어 낸다는 것을 알 수 있다. 반면에 플립플롭 사전 패킹은 전체적으로 좋은 결과를 냈지만 배치 패턴에 영향을 많이 준다는 것을 확인할 수 있다.

수행시간은 조건에 관계없이 거의 동일하며 지면 관계상 표시하지 않았다.

Ⅳ. 결 론

단순한 가상 모델이 아닌 상용 FPGA의 아키텍처에 적용되는 분석적 배치 및 배치 적법화에서 실제로 발생되는 이슈들을 도출하여 대안을 제시하고 그 효과를 평가하였다. 먼저, 코어 사용률이 낮은 FPGA에서 발생할 수 있는 무게 중심 변위를 최소화하는 함수를 분석적 배치의 목적 함수에 추가하였다. 또한, 배치 밀도 평가에서 셀 종류별로 별도의 밀도 행렬을 사용하는 다층 분석과 자원이 매우 한정된 블록의 조기 고정 방안을 제안하였다. 그 밖에, 자원 공유 때문에 발생하는 호환성 문제를 개선하기 위한 플립플롭 사전 패킹도 제안하였다.

제안된 기법은 상용 FPGA 아키텍처를 모델링하고 수정 개선할 수 있는 K-FPGA 패브릭 평가 툴킷을 근간으로 구현되었으며 12개의 실용 예제를 통해 기존 방식에 비해 평균적으로 배선 길이 22%, 슬라이스 사용량 5%를 감축하는 효과를 확인하였다.

또한 배치 적법화에서는 미처 적법 위치를 찾지 못한 셀들을 외곽으로 밀어내기보다 사전에 빈들의 밀도 균형을 맞추는 절차가 필요하고, 플립플롭 사전 패킹에는 인접성 고려가 요구되는 등 추가적인 개선 여지를 확인하였다. 이 밖에 실용성을 위해서는 무엇보다도 타이밍 구동 형태로 진화되어야 한다.

REFERENCES

[1] *ABC: A System for Sequential Synthesis and Verification.* Berkeley Logic Synthesis and Verification Group, http://www.eecs.berkeley.edu/~alanmi/abc/abc.html, October, 2007.

[2] V. Betz and J. Rose, "VPR: A New Packing, Placement And Routing Tool For FPGA Research," in *Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications*. pp.213 - 222, 1997.

[3] K. Kim, "Fabric Mapping and Placement of Field Programmable Stateful Logic Array," *Journal of the IEEK,* vol. 49, no. 12, pp. 1067-1076, December, 2012.

[4] N. Steiner, A. Wood, H. Shojaei, J. Couch, P. Athanas, M. French, "Torc: Towards Open-Source Tool Flow," in *Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pp.41-44, February, 2010.

[5] C. Lavin, M. Padilla, J. Lamprecht, P. Lundrigan, B. Nelson, and B. Hutchings, "RapidSmith: Do-It-Yourself CAD Tools for Xilinx FPGAs" in *Proceedings of the 21st International Workshop on Field-Programmable Logic and Applications*, pp.349-355, September, 2011.

[6] K. Kim, “Evaluation Toolkit for K-FPGA Fabric Architectures,” *Journal of the IEEK*, vol. 49-SD, no. 4, pp.157-167, April, 2012.
[7] *Spartan-3 Generation FPGA User Guide*, UG331, v1.6, Xilinx Inc., December 3, 2009.
[8] W.C. Naylor, R. Donelly, and L. Sha, “Non-Linear Optimization System and Method for Wire Length and delay Optimization for an Automatic Electric Circuit Placer,” *US Patent* 6301693, October 2001.
[9] J. Cong and G. Luo, “Highly Efficient Gradient Computation for Density-Constrained Analytical Placement Methods,” *Proc. of the International Symposium on Physical Design*, pp. 39-45, April, 2008.
[10] D.J.C. MacKay, “MacOpt-a Nippy Wee Optimizer,” http://www.inference.phy.cam.ac.uk/mackay/c/macopt.html, June, 2004.
[11] X. Song, “Smoothing Method for Minimax Problems,” *Computational Optimization and Applications*, Kluwer Academic Publishers, 20, pp.267 - 279, 2001.
[12] *Xilinx Design Language Version 1.6*, Xilinx, Inc., Xilinx ISE 6.1i Documentation in ise6.1i/help/data /xdl, July 2000.

저 자 소 개

김 교 선(정회원)
1986년 연세대학교 전자공학과 학사 졸업.
1988년 연세대학교 전자공학과 석사 졸업.
1998년 Ph.D. Department of Electrical & Computer Engineering, University of Massachusetts, Amherst, U.S.A
1988년～2003년 삼성전자 CAE Center 주임, 선임, 책임, 수석 연구원
현재 인천대학교 공과대학 전자공학과 교수
<주관심분야 : 상위수준합성, 논리합성, 배치, Reconfigurable Computation, Fault-Tolerance, Embedded Systems, Low-Power Design, Nanoelectronic Architectures>