Journal of The Institute of Electronics and Information Engineers Vol.54, NO.7, July 2017 https://doi.org/10.5573/ieie.2017.54.7.21
ISSN 2287-5026(Print) / ISSN 2288-159X(Online)

논문 2017-54-7-3

타이밍 구동 FPGA 분석적 배치

(Timing Driven Analytic Placement for FPGAs)

김 교 선*

(Kyosun Kim©)

요 약

FPGA 배치 툴 연구는 학계에서도 단순한 가상 아키텍처 모델 가정에서 벗어나 상용 툴처럼 캐리체인이나 광폭함수 멀티플렉서, 메모리/승산기 블록 등의 성능 및 밀도를 향상시키는 소자들을 포함하는 현실적인 모델을 적용하기 시작하였다. 이 때 발생하는 실제적 이슈들을 다룬 사전 패킹, 다층 밀도 분석 등의 기법이 초기 분석적 배치 (Analytic Placement)에 적용되어 밀도를 분산시키면서 배선 길이를 효과적으로 최소화한 연구가 앞서 발표된 바 있다. 더 나아가 궁극적으로는 타이밍을 최적화해야하기 때문에 많은 연구에서는 타이밍 제약 조건을 만족시키기 위한 기법들이 제시되고 있다. 그러나 초기 배치 후 진행되는 배치 적법화 및 배치 개선에서 주로 적용될 뿐 분석적 배치에서 이러한 타이밍 기법을 적용한 사례는 거의 없다. 본 논문에서는 사전 패킹 및 다층 밀도 분석 등의 기법이 구현된 기존 분석적 배치에 타이밍 제약 조건 위반을 검출하고 이를 최소화하는 기법을 결합하는 방안을 소개한다. 먼저 정적 타이밍 검증기를 집적하여 배선 길이가 최소화된 기존 배치 결과의 타이밍을 검사해 보았으며 위반을 감소시키기 위해 신호 도착 시간 (Arrival Time)을 최소화하는 함수를 분석적 배치의 목적 함수에 추가하였다. 이 때 각 클록마다 주기가 다를 수 있기 때문에 각 클록별로 함수를 따로 계산해 합산하는 방안이 제안되었다. 또한, 위반이 없는 클록 도메인의 신호 경로들도 불필요하게 단축될 수 있기 때문에 음수 슬랙 (Negative Slack)을 계산하여 이를 최소화하는 함수를 추가로 제안하여 비교하였다. 영역 분할 기법 (Partitioning)을 기반으로 배선 길이를 최소화하는 기존 배치 적법화를 그대로 사용한 후 타이밍 검증을 통해 초기 분석적 배치 단계에서 타이밍 개선 효과를 분석하였다. 배치 적법화 시 추가적인 타이밍 최적화 기법이 사용되지 않았기 때문에 타이밍 개선이 있다면 이것은 전적으로 분석적 배치의 목적 함수 개선에 의한 효과이다. 12개 실용예제에 대해 실험한 결과, 목적 함수에 도착 시간 함수가 적용되었을 때 그렇지 않았을 때보다 최악 음수 슬랙 (Worst Negative Slack)이 평균 약 15% 정도 감소되었으며 음수 슬랙 함수가 적용되었을 때 이보다 약 6% 정도 추가로 더 감소됨을 확인하였다.

Abstract

Practical models for FPGA architectures which include performance- and/or density-enhancing components such as carry chains, wide function multiplexers, and memory/multiplier blocks are being applied to academic FPGA placement tools which used to rely on simple imaginary models. Previously the techniques such as pre-packing and multi-layer density analysis are proposed to remedy issues related to such practical models, and the wire length is effectively minimized during initial analytic placement. Since timing should be optimized rather than wire length, most previous work takes into account the timing constraints. However, instead of the initial analytic placement, the timing-driven techniques are mostly applied to subsequent steps such as placement legalization and iterative improvement. This paper incorporates the timing driven techniques, which check if the placement meets the timing constraints given in the standard SDC format, and minimize the detected violations, with the existing analytic placer which implements pre-packing and multi-layer density analysis. First of all, a static timing analyzer has been used to check the timing of the wire-length minimized placement results. In order to minimize the detected violations, a function to minimize the largest arrival time at end points is added to the objective function of the analytic placer. Since each clock has a different period, the function is proposed to be evaluated for each clock, and added to the objective function. Since this function can unnecessarily reduce the unviolated paths, a new function which calculates and minimizes the largest negative slack at end points is also proposed, and compared. Since the existing legalization which is non-timing driven is used before the timing analysis, any improvement on timing is entirely due to the functions added to the objective function. The experiments on twelve industrial examples show that the minimum arrival time function improves the worst negative slack by 15% on average whereas the minimum worst negative slack function improves the negative slacks by additional 6% on average.

Keywords : FPGA, Analytic Placement, Timing Driven, Arrival Time, Worst Negative Slack

* 정회원, 인천대학교 (Incheon National University)
© Corresponding Author (E-mail : kkim@inu.ac.kr)
※ 이 논문은 인천대학교 2015년도 자체연구비 지원에 의하여 연구되었음.
Received ; April 12, 2017 Revised ; May 16, 2017 Accepted ; June 8, 2017

Ⅰ. 서 론

상용 FPGA (Field Programmable Gate Array) 구조는 개발사의 전유물이고 학계에서 수집되는 정보는 매우 제약적이어서 CAD 툴을 개발하는 연구가 단순한 가상 FPGA 구조[1~2]를 가정하여 왔으나 최근 학계에도 다행히도 Xilinx Spartan/Virtex 계통의 FPGA를 중심으로 툴 개발 환경이 확보되고 있다. USC 그룹의 Torc[3] 및 BYU 그룹의 RapidSmith[4]가 툴킷 형태로 공개된 것이 대표 사례이다. 두 툴킷 모두 Xilinx 사의 XDL 파일 형식을 근간으로 FPGA 구조 데이터베이스를 구축하고 C++/Java API (Application Programming Interface)를 제공함으로써 설계 입력, 논리 합성, 패킹, 배치, 배선, 아키텍처 뷰어 등을 플러그-인 할 수 있도록 하고 있다. 국내에서도 한국형 K-FPGA 개발을 국책 과제[5]로 진행하였으며 툴킷이 개발되었다. 상용 FPGA 구조를 모델링할 뿐만 아니라 이의 변형 또는 새로운 구조를 모델링하여 데이터베이스를 구축할 수 있고 이를 기반으로 논리합성, 배치, 배선, 뷰어, 이종 툴과의 인터페이스 등이 이미 플러그-인 되어 있다.

최근 상용 FPGA 구조를 정확히 모델링할 수 있는 K-FPGA 툴킷[5] 상에서 초기 배치 (Initial Placement)에 사용되는 분석적 배치 (Analytic Placement)[6]와 배치 적법화 (Legalization)에 상용 구조 관련된 이슈들에 대한 해법으로 사전 패킹 (Pre-Packing) 및 배치 밀도 분산 다층화 (Multi-Layer Density Distribution)를 제안하여 효과적으로 배선 길이를 최적화한 바 있다[7]. 더 나아가 궁극적으로는 타이밍을 최적화해야하기 때문에 대부분의 기존 연구에서는 타이밍 제약 조건을 만족시키기 위한 기법을 제시하고 있다. 그러나 초기 배치 후 진행되는 배치 적법화 및 배치 개선에서 주로 적용될 뿐 분석적 배치에서 이러한 타이밍 기법을 적용한 사례는 거의 없다. GPlace[8]는 배선 길이를 별 모양으로 계산하는 스타 플러스 (Star+) 모델을 사용하고 UTPlaceF[9]는 유클리드 거리를 제곱한 모델을 사용하지만 초기 배치 시 신호 경로 타이밍을 최적화하지는 않는다. 시높시스 사 특허[10]만이 초기 배치 시에 도착 시간을 최소화하는 함수를 소개하고 있다. UTPlaceF[9]는 K-FPGA[5, 7]처럼 상용 아키텍처의 성능 및 밀도 향상 소자를 고려한 패킹 문제를 다루고 있다.

본 논문에서는 사전 패킹 및 다층 밀도 분석 등의 기법이 구현된 기존 분석적 배치에서 타이밍 제약 조건 위반을 검출하고 이를 최소화하는 기법을 결합하는 방안을 소개한다. 신호 도착 시간 (Arrival Time)을 최소화하는 함수를 분석적 배치의 목적 함수에 추가하되 각 클록마다 주기가 다를 수 있기 때문에 각 클록별로 함수를 따로 계산해 합산하는 방안이 제안되었다. 또한, 위반이 없는 클록 도메인의 신호 경로들도 불필요하게 단축될 수 있기 때문에 슬랙 (Negative Slack)을 계산하여 슬랙이 음수일 때만 최소화하는 함수를 추가로 제안하여 비교하였다. 배치 적법화 후 초기 분석적 배치 단계에서 타이밍 개선 효과를 분석하였다. 배치 적법화 시 추가적인 타이밍 최적화 기법이 사용되지 않았기 때문에 타이밍 개선이 있다면 이것은 전적으로 분석적 배치의 목적 함수 개선에 의한 효과이다.

본 논문은 먼저 Xilinx Spartan 구조 및 툴 체인 사용 절차를 2장 본론 1절 및 2절에서 소개하고 3절에서 분석적 배치에 추가된 타이밍 함수들을 기술한다. 3장 실험에서 12개 실용 예제에 대해 적용한 결과를 제시하고 고찰한 후 마지막으로 결론을 맺는다.

Ⅱ. 본 론

1. 상용 FPGA 패브릭 (Fabric) 구조

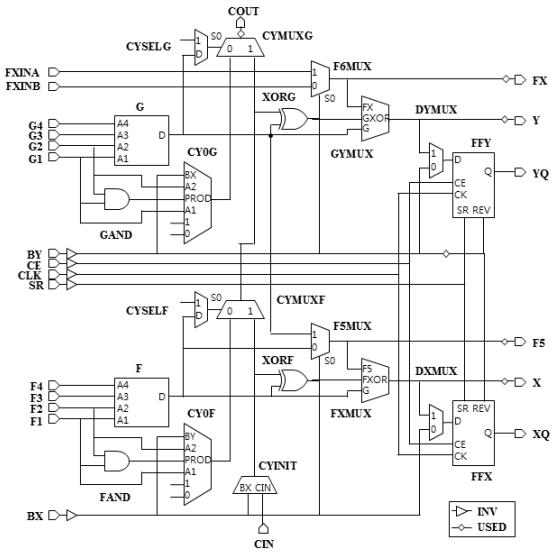


그림 1. Xilinx Spartan Ⅲ 슬라이스 (SLICEL) 구조
Fig. 1. Slice structure (SLICEL) of the Xilinx Spartan Ⅲ.

그림 1에 Xilinx Spartan Ⅲ[11] 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.

2. FPGA 설계 툴 체인 사용 절차

그림 2에 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.

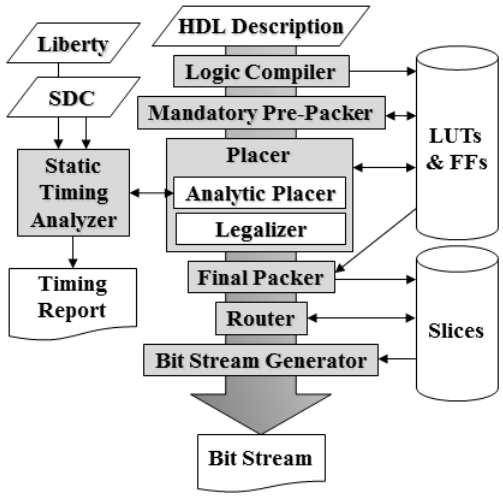


그림 2. FPGA 설계 툴 체인 사용 절차
Fig. 2. FPGA design flow using the tool chain.

흔히, 배치는 분석적 배치 및 적법화로 구성되며 최종 패킹 (Final Packer)이 LUT과 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 사용되는 논리 단위 셀들은 멀티플렉서 및 전용 게이트들 간의 인접성을 만족시키기 위해 상대적 배치 위치가 미리 결정되며 이를 계속 유지해야 하므로 필수 전처리 패킹 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.

타이밍의 검증 및 최적화를 위해 정적 타이밍 검증기 (Static Timing Analyzer)가 필요하며 K-FPGA 과제에서 구현된 것은 표준화된 Liberty 형식으로 기술된 라이브러리 프리미티브 셀들의 지연 정보를 사용하여 회로의 타이밍을 계산하고 표준화된 SDC (Synthesis Design Constraint) 형식으로 기술된 제약 조건을 검사할 수 있다. 논리 합성, 배치 배선 등 각 설계 단계 후 타이밍을 검증하도록 개발되어 있었는데 이것을 분석적 배치에 집적하여 타이밍 최적화에 응용하였다.

3. 분석적 배치 (Analytic Placement)

분석적 배치는 비선형 최적화 기법[6~10]을 이용하며 다음과 같은 목적함수를 사용하는 Conjugate Gradient (CG) Solver[12]가 응용된다.

$$\begin{aligned} &objective \\ &= \alpha_l\, length + \alpha_t\, timing + \alpha_d\, density \\ &\quad + \alpha_b\, barrier + \alpha_c\, cogd \end{aligned} \tag{1}$$

목적함수는 배선 길이 (length), 타이밍 (timing), 배치 밀도 (density), 장벽 (barrier), 그리고 무게 중심 변위 (cogd : Center of Gravity Displacement) 함수로 구성된다. 목적 함수 내 다섯 가지 함수의 영향을 조정하기 위해 가중치 α_l, α_t, α_d, α_b, α_c가 사용되었다. 이중 타이밍 함수를 제외한 모든 함수들은 이미 선행 연구[7]에서 제시되어 상세히 설명되어 있다.

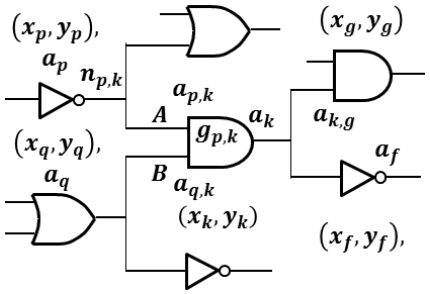


그림 3. 게이트의 배치 좌표, 지연 시간 및 도착 시간
Fig. 3. Placement coordinates, delays, and arrival times of gates.

타이밍 함수를 설명하기 위해서 먼저 그림 3과 같은 게이트 수준 회로를 살펴보자. 중앙에 AND 게이트 k가 있으며 입력 A에 인버터 p의 출력이 연결되어 있다. 이

인버터와 입력 B에 연결된 OR 게이트 q를 게이트 k의 팬-인 (Fan-In) 게이트라 하며 게이트 f와 g는 게이트 k의 팬-아웃 (Fan-Out) 게이트라 한다. 게이트 p의 출력까지의 신호 도착시간 (Arrival Time)은 a_p이다. 여기부터 게이트 k의 입력 A까지 배선 지연은 $n_{p,k}$이며 도착시간은 $a_{p,k}$이다. 입력 A에서 출력까지 게이트 지연은 $g_{p,k}$이며 도착시간은 a_k이다. 각 게이트마다 배치 좌표가 있으며 게이트 k의 좌표는 (x_k, y_k)이다. 타이밍 함수는 다음과 같이 정의할 수 있다.

$$timing = \sum_{c \in cd0} A_c \tag{2}$$

$$A_c = \max_{t \in ep(c)} \boldsymbol{a_t} = \alpha \log \sum_{t \in ep(c)} e^{\frac{a_t}{\alpha}} \tag{3}$$

$$\begin{aligned} \boldsymbol{a_k} &= \max_{p \in fanin(k)} \{\boldsymbol{a_{p,k}} + \boldsymbol{g_{p,k}}\} \\ &= \alpha \log \sum_{p \in fanin(k)} e^{\frac{a_{p,k}+g_{p,k}}{\alpha}} \end{aligned} \tag{4}$$

$$\boldsymbol{a_{p,k}} = \boldsymbol{a_p} + \boldsymbol{n_{p,k}} \tag{5}$$

$$\begin{aligned} \boldsymbol{n_{p,k}} &= \tau_w\{|x_p - x_k| + |y_p - y_k|\} \\ &= \tau_w\gamma \begin{Bmatrix} \log\left(e^{\frac{x_p}{\gamma}} + e^{\frac{x_k}{\gamma}}\right) + \log\left(e^{-\frac{x_p}{\gamma}} + e^{-\frac{x_k}{\gamma}}\right) \\ + \log\left(e^{\frac{y_p}{\gamma}} + e^{\frac{y_k}{\gamma}}\right) + \log\left(e^{-\frac{y_p}{\gamma}} + e^{-\frac{y_k}{\gamma}}\right) \end{Bmatrix} \end{aligned} \tag{6}$$

먼저, 식 (2)에서 cd()는 클록 도메인 집합이며 식 (3)에서 ep(c)는 클록 도메인 c의 타이밍 끝점 (End Point) 집합, a_t는 끝점 t의 도착시간이다. 따라서, A_c는 클록 도메인 c 안의 a_t들 중 최대값이며 지수-합-로그 (Log-Sum-Exponent) 함수로 스무딩 (Smoothing)되었다. 끝점이 아닌 게이트들에 대해 a_k는 식 (4)와 같이 출력까지 가장 늦게 도착하는 입력 신호의 도착 시간이 되며 스무딩 되어 있다. $a_{p,k}$는 식 (5)와 같이 게이트 p 출력의 도착시간 a_p와 게이트 k까지의 배선 지연 $n_{p,k}$의 합이 된다. 식 (6)과 같이 $n_{p,k}$는 맨해튼 거리에 비례하며 역시 스무딩 되어있다.

비선형 최적화 시 좌표 변화가 목적 함수에 미치는 영향을 예측하기 위해 편미분 함수를 계산해야 하는데 특히 타이밍 함수에 대한 편미분 함수는 다음과 같이 연쇄법칙 (Chain Rule)에 의해 단계적으로 계산될 수 있다.

먼저 타이밍 끝점에서 도착 시간에 대한 타이밍 함수의 편미분은 식 (7)처럼 계산된다. 그러나 그밖에 게이트들에 대해서는 모두 식 (8)이 적용된다. 여기서 out(k)는 게이트 k의 팬-아웃 게이트 집합이고 in(k)는 팬-인 게이트 집합이다.

$$\frac{\partial timing}{\partial \boldsymbol{a_t}} = \frac{\partial \boldsymbol{A_c}}{\partial \boldsymbol{a_t}} = \frac{e^{\frac{a_t - A_c}{\alpha}}}{\sum_{i \in ep(c)} e^{\frac{a_i - A_c}{\alpha}}} \tag{7}$$

$$\frac{\partial \boldsymbol{A_c}}{\partial \boldsymbol{a_k}} = \sum_{f \in out(k)} \frac{\partial \boldsymbol{A_c}}{\partial \boldsymbol{a_f}} \frac{\partial \boldsymbol{a_f}}{\partial \boldsymbol{a_k}} = \sum_{f \in out(k)} \frac{\partial \boldsymbol{A_c}}{\partial \boldsymbol{a_f}} \frac{\partial \boldsymbol{a_f}}{\partial \boldsymbol{n_{k,f}}} \tag{8}$$

$$\frac{\partial \boldsymbol{a_k}}{\partial \boldsymbol{n_{p,k}}} = \frac{e^{\frac{a_{p,k}+g_{p,k}}{\alpha}}}{\sum_{i \in in(k)} e^{\frac{a_{i,k}+g_{i,k}}{\alpha}}} \tag{9}$$

$$\frac{\partial \boldsymbol{n_{p,k}}}{\partial x_k} = \tau_w \left\{ \frac{e^{\frac{x_k - x_{max}}{\gamma}}}{e^{\frac{x_p - x_{max}}{\gamma}} + e^{\frac{x_k - x_{max}}{\gamma}}} - \frac{e^{-\frac{x_k}{\gamma}}}{e^{-\frac{x_p}{\gamma}} + e^{-\frac{x_k}{\gamma}}} \right\} \tag{10}$$

$$\frac{\partial \boldsymbol{n_{p,k}}}{\partial y_k} = \tau_w \left\{ \frac{e^{\frac{y_k - y_{max}}{\gamma}}}{e^{\frac{y_p - y_{max}}{\gamma}} + e^{\frac{y_k - y_{max}}{\gamma}}} - \frac{e^{-\frac{y_k}{\gamma}}}{e^{-\frac{y_p}{\gamma}} + e^{-\frac{y_k}{\gamma}}} \right\} \tag{11}$$

$$\frac{\partial timing}{\partial \boldsymbol{x_k}} = \frac{\partial \boldsymbol{A_c}}{\partial \boldsymbol{a_k}} \sum_{p \in in(k)} \frac{\partial \boldsymbol{a_k}}{\partial \boldsymbol{n_{p,k}}} \frac{\partial \boldsymbol{n_{p,k}}}{\partial \boldsymbol{x_k}} \tag{12}$$

$$\frac{\partial timing}{\partial \boldsymbol{y_k}} = \frac{\partial \boldsymbol{A_c}}{\partial \boldsymbol{a_k}} \sum_{p \in in(k)} \frac{\partial \boldsymbol{a_k}}{\partial \boldsymbol{n_{p,k}}} \frac{\partial \boldsymbol{n_{p,k}}}{\partial \boldsymbol{y_k}} \tag{13}$$

타이밍 검증은 먼저 배치 좌표로부터 배선 지연 계산부터 하게 되는데 이 때 식 (10)과 (11)도 같이 계산된다. 이어 타이밍 끝점부터 시작점으로 재귀적 함수 호출 (Recursive Function Call) 방식으로 너비우선탐색 (Breadth First Search)을 하면서 도착 시간 (a_k, $a_{p,k}$) 계산을 하는데 이 때, 식 (9)도 계산된다. 식 (7)은 도착 시간 계산이 모두 끝난 다음 모든 끝점에 대해 계산된다. 다음은 타이밍 시작점부터 역시 재귀 함수 호출 방식으로 역방향 (Reverse) 너비우선탐색을 하면서 타이밍 제약 조건으로 주어진 클록의 주기를 기준으로 한 요구 시간 (Required Time) 계산을 하는데 이 때 식 (8)이 같이 계산된다. 요구 시간과 도착 시간의 차가 슬랙 (Slack)이며 음수가 되면 타이밍 제약을 위반한 것이다. 정적 타이밍 검증기는 위반된 경로를 클록 도메인별로 슬랙의 크기에 따라 정렬하여 리포트 할 수 있다. 타이밍 끝점에서 음수 슬랙 중 가장 큰 값을 가진 것을 최악 음수 슬랙, 즉 WNS (Worst Negative Slack)라 하고 이들의 합을 TNS (Total Negative Slack)라 하며 타이밍 만족도를 평가하는 척도로 사용된다.

비선형 최적화에서 목적 함수의 편미분이 다시 계산될 때마다 타이밍 검증이 반복되며 타이밍 검증이 끝나면 식 (12)와 (13)이 계산되어 가중치가 곱해진 후 목적 함수 편미분에 더해진다.

지금까지 소개된 타이밍 함수와 편미분 함수는 대부분 미국 시높시스 사의 특허[10]에서 인용한 것이나 (1) 배선 모델을 FPGA 구조에 적합한 맨해튼 거리 계산식으로 바꾸고 (2) 다중 클록이 사용될 때 최대 도착시간을 클록별로 최소화하도록 수정한 것이다. 또한, (3) 사전 패킹, 다층 밀도 분석 등 상용 FPGA 아키텍처 이슈 관련 기술과 복합적으로 적용되며, (4) 타이밍 함수의 효과가 초기 분석적 배치 결과 단독으로 평가된다는 점도 다르다. 그럼에도 불구하고 이 모델은 슬랙이 양수일 경우에도 계속해서 타이밍을 최적화하기 때문에 아직 최적화되지 못한 다른 클록 도메인의 타이밍 개선을 방해할 수 있다는 단점이 있다.

이를 개선하기 위해 본 논문은 도착 시간 대신 음수 슬랙을 최소화하는 타이밍 함수를 제안한다. 이는 앞서 소개한 타이밍 목적 함수에서 식 (2)와 (3)을 다음과 같은 식 (2-1)과 (3-1)로 대체함으로써 간단히 구성할 수 있다. 즉, 식 (2)(3)에 기술된 클록별 도착 시간 최대치의 합을 식 (2-1)(3-1)과 같이 음수 슬랙의 최대치로 치환하면 새로운 타이밍 목적함수를 얻을 수 있다. 식 (4) 이후의 식들은 추가적 변형 필요 없이 식 (2-1)(3-1)과 자연스럽게 체인 형태로 이어진다.

$$\boldsymbol{timing} = \max_{t \in ep()} \boldsymbol{v_t} = \alpha \log \sum_{t \in ep()} e^{\frac{v_t}{\alpha}} \qquad (2\text{-}1)$$

$$\boldsymbol{v_t} = \max(\boldsymbol{a_t} - \boldsymbol{r_t}, \boldsymbol{0}) = \alpha \log\left(e^{\frac{a_t - r_t}{\alpha}} + 1\right) \qquad (3\text{-}1)$$

r_t는 타이밍 끝점 t에서 요구 시간이며 신호 경로상의 각 게이트의 도착 시간에 영향을 미치지 않으므로 상수로 생각할 수 있다. v_t는 타이밍 끝점 t에서의 위반 정도, 즉 음수 슬랙이다. 슬랙이 양수이면 0이 되며 슬랙이 음수이면 부호를 바꾸어 양수가 되도록 한 것이다. 이 함수는 식 (3-1)과 같이 최대값 함수로 표현될 수 있으며 지수-합-로그 함수로 쉽게 스무딩 된다. 따라서 이 타이밍 함수는 타이밍 위반이 있는 끝점에 도달하는 경로만 최적화하도록 하며 그 경로들 중 최악 경로를 우선적으로 개선하도록 한다. 타이밍 함수의 미분 수식 역시 다음과 같이 앞서 소개한 수식 (7)만 수정하면 얻을 수 있다.

클록 도메인마다 타이밍 만족 정도가 다르고 그 편차가 클 경우 이 WNS 타이밍 함수가 개선 정도를 더 향상시킬 수 있을 것으로 기대된다.

$$\frac{\partial \boldsymbol{timing}}{\partial \boldsymbol{v_t}} = \frac{e^{\frac{v_t - v_{max}}{\alpha}}}{\sum_{i \in ep()} e^{\frac{v_i - v_{max}}{\alpha}}} \qquad (7\text{-}1)$$

$$\frac{\partial \boldsymbol{v_t}}{\partial \boldsymbol{a_t}} = \frac{e^{\frac{a_t - r_t}{\alpha}}}{e^{\frac{a_t - r_t}{\alpha}} + 1} \qquad (7\text{-}2)$$

$$\frac{\partial \boldsymbol{timing}}{\partial \boldsymbol{a_t}} = \frac{\partial \boldsymbol{timing}}{\partial \boldsymbol{v_t}} \frac{\partial \boldsymbol{v_t}}{\partial \boldsymbol{a_t}} \qquad (7\text{-}3)$$

Ⅲ. 실 험

제안된 기법들의 효용성을 확인하기 위해 기존 K-FPGA 분석적 배치 모듈[7]에 정적 타이밍 검증기를 집적하였으며 목적함수에 제안된 두 가지 타이밍 함수를 포함하도록 수정하였다. 분석적 배치에서는 가중치를 사용하여 배선 길이, 도착시간 타이밍, WNS 타이밍, 밀도, 장벽, 무게 중심 변위 함수를 목적 함수에 선택적으로 포함시키거나 그 영향을 조절할 수 있다. 또한, 호환성 있는 플립플롭 쌍의 사전 패킹, 블록 메모리의 조기 고정 기능도 선택적으로 적용할 수 있다. 쉘과 유사한 환경에서 스크립트로 실행시킬 수 있도록 각 기능 실행 및 파라미터 조건 변경을 위한 명령들을 구현해 놓고 있으며 배치 결과를 그래픽으로 확인할 수 있다.

표 1. 12개 예제의 회로 규모 및 소자 사용 프로파일
Table1. 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

Xilinx Spartan III XC3S1000 상용 마스터와 다양한 프로파일의 12가지 예제를 통해 배치 실험을 수행하였다. 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. 플립플롭이나 LUT 뿐만 아니라 여러 가지 효율 개선 소자들이 각기 다른 형태와 규모로 분포되어 있는 실용 예제가 선별되었음을 알 수 있다. 또한 이들은 기존 연구[7]에서 사용되었던 예제를 첨삭 없이 그대로 사용한 것이다.

표 2. 12개 예제의 복잡도
Table2. 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

표 3. 그리드 해상도, 스무딩, 함수 가중치 파라미터
Table3. Parameters for grid resolution, smoothing, and function weights.

조건	회	slices /grid	γ	r	α_l	α_t	α_d	α_b	α_c
no timing	1	4x4	1.5	3	2	0	1	4	20
	2	2x2	1.5	3.5	1	0	2	2	20
timing driven	1	4x4	1.5	3	0.8	8	1	4	20
	2	2x2	1.5	3.5	0.8	8	2	2	20

세 가지 조건의 목적 함수 즉, (1) 타이밍 함수를 포함하지 않은 (no timing), (2) 클록별 도착시간 최소화 함수를 포함한 (arrival), 그리고 (3) 최악 음수 슬랙 최소화 함수를 포함한 (wns) 목적 함수를 각각 사용하여 분석적 배치와 배치 적법화를 교대로 2회 실시하였으며 가중치 등의 파라미터들은 표 3에 요약한 바와 같다.

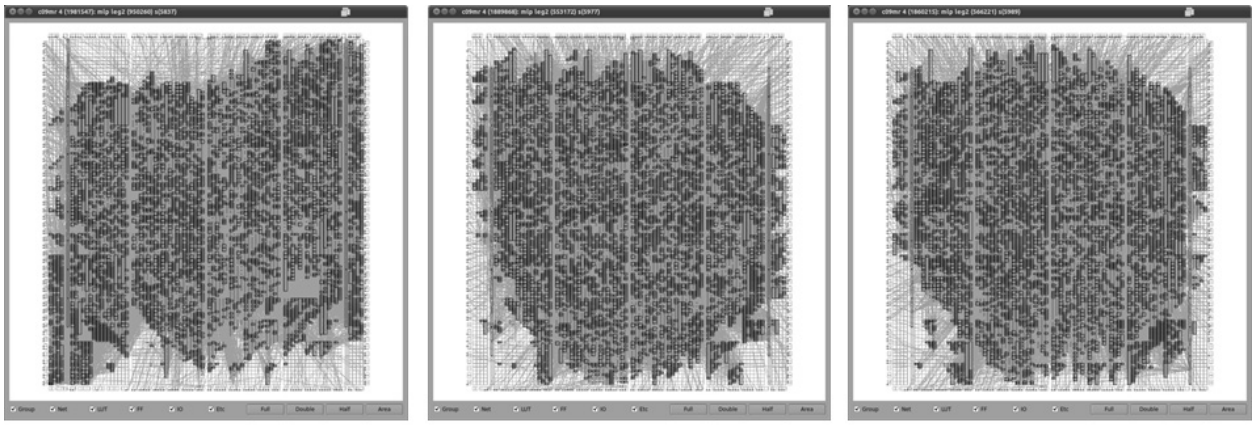
조건 (1)은 선행 연구[7] 실험 조건과 동일하며 조건 (2), (3)과 같이 타이밍 함수가 포함될 경우에는 배선 길이 함수 가중치를 낮추어 밀도를 높이는 인장력을 완화하는 것이 더 효과적임이 예비 실험을 통해 확인되었다. 반면, 타이밍 함수는 배선 길이 함수와는 달리 모든 배선에 동시에 영향을 미치지 않고 특정 신호 경로의 시간 지연을 집중적으로 개선하기 때문에 가중치를 크게 주어도 힘의 균형이 쉽게 무너지지 않는다는 것도 확인하였다. 두 타이밍 함수 (arrival, wns)에 적용 조건은 동일 (timing driven)하며 모든 예제에 동일하게 적용하였다. 배치 적법화에서 타이밍 구동 기법을 적용하지 않았기 때문에 결과의 차이는 순전히 분석적 초기 배치의 목적함수에 타이밍 함수의 적용 여부가 만들어 낸 것이라 할 수 있다.

12개 예제를 세 조건에 따라 배치한 결과를 타이밍 분석하였으며 표 4에 정리하였다. 각각에 대해 최악 음수 슬랙 (WNS)과 음수 슬랙 총합 (TNS)을 비교하였으며 클록별 도착 시간 최소화 함수를 포함시켰을 때 이를 제외했을 때보다 WNS는 평균 약 15% 감소했으며 TNS는 약 13% 감소했다. 모든 배선 지연을 다 같이 단축시키는 것보다 클록 도메인별로 가장 긴 경로만을 단축시키는 것이 더 효율적임을 확인할 수 있다. 최악 음수 슬랙 최소화 함수를 적용했을 때는 이보다 WNS가 추가적으로 평균 약 6% 더 감소했으며 TNS도 약 10% 더 감소했다.

표 4. 12개 예제 초기 분석적 배치 결과 타이밍 함수별 타이밍 분석
Table4. Timing analysis of twelve examples for each timing function after initial analytic placement.

name	no timing		arrival		wns	
	WNS	TNS	WNS	TNS	WNS	TNS
c01bp	-6.64	-344.2	-5.91	-408.7	-5.96	-482.1
c02cc	-5.19	-14.4	-4.40	-19.6	-4.36	-20.3
c03dr	-2.50	-98.6	-2.66	-93.0	-1.73	-151.6
c04em	-4.59	-477.6	-4.45	-372.1	-4.37	-384.7
c05fp	-7.48	-1641.9	-7.39	-1625.0	-7.34	-1613.3
c06ie	-1.50	-37.0	-0.86	-21.3	-1.20	-32.3
c07im	-4.23	-273.8	-5.35	-331.4	-4.05	-231.0
c08ip	-3.67	-421.3	-4.06	-419.9	-4.39	-409.9
c09mr	-4.26	-109.2	-3.35	-114.8	-2.07	-92.0
c10ne	-4.76	-305.8	-4.49	-357.4	-4.70	-314.2
c11pc	-3.81	-19.4	-3.75	-24.6	-3.75	-20.3
c12sw	-18.88	-3062.6	-10.88	-2111.1	-10.28	-1537.2
avg	-5.63	-567.1	-4.79	-491.6	-4.52	-440.8
%			14.77	13.3	5.82	10.3

타이밍 위반 여부에 상관없이 무조건 도착 시간을 단축시키기보다 위반된 경로의 도착 시간만 단축시킴으로서 그 효율을 더 높일 수 있음을 확인할 수 있다. 참고적으로 세 조건에 따른 총 배선 길이를 비교하여 표 5에 정리하였다. 배선 길이는 도착 시간 최소화 함수 적



(a) no timing (b) minimum arrival time (c) minimum slack

그림 4. c09mr 회로의 3가지 목적 함수에 대한 배치 결과
Fig. 4. Placement results of c09mr for three objective functions.

용 시 4% 정도 증가했다가 최악 음수 슬랙 함수 적용 시 다시 이보다 2% 정도 감소하는 정도로 크게 변화하지 않았으며 최악 경로 중심으로 국부적인 배치 개선이 있었다고 판단할 수 있다.

표 5. 총 배선 길이 비교
Table5. Comparison of total net lengths.

name	no timing	arrival	wns
c01bp	66896900	75034100	77498000
c02cc	75793900	80686500	77633300
c03dr	85990500	96383800	96359700
c04em	74413500	80595300	79867000
c05fp	63159400	58235100	59636400
c06ie	58831600	62152700	62405700
c07im	62942700	70096900	69283400
c08ip	133344900	138650300	144393300
c09mr	198154700	188986800	186021500
c10ne	98905300	114416100	113454900
c11pc	34465900	35243700	34670200
c12sw	225721100	227070000	207287600
avg	98218367	102295942	100709250
%		4.15	-1.55

타이밍 목적 함수가 레이아웃 패턴에 미치는 영향을 분석하기 위해 12개 예제 중 비교적 다양한 소자를 포함하고 있으며 뚜렷한 개선 효과를 보인 c09mr을 선택하였다. 이 c09mr에 대해 세 가지 타이밍 함수 적용 조건에 따른 배치 결과 레이아웃을 그림 4에 비교하였다. 그림 4 (a)와 나머지 (b), (c)를 비교하면 타이밍 함수가 목적 함수에 추가됨에 따라 배치 구조가 대폭 바뀔 수도 있음을 알 수 있다. 또한, 두 타이밍 함수 적용 결과 (b)와 (c)를 비교하면 유사한 배치 구조를 나타내지만 국부적인 패턴이 다름을 알 수 있다.

Ⅳ. 감사의 글

K-FPGA 과제에서 개발된 정적 타이밍 검증 모듈을 제공하여 본 과제에 적용할 수 있도록 허락해 주신 ㈜엔타시스 사 오성환 대표에 감사드린다.

Ⅴ. 결 론

분석적 초기 배치에서는 원래 배선 길이를 최소화하는 동시에 배치 밀도가 분산되도록 하는 주요 기능에 국한하고 신호 경로의 타이밍 최적화는 주로 배치 적법화나 반복적 배치 개선 단계에서 진행되어왔다. 본 논문에서는 정적 타이밍 검증기를 분석적 초기 배치에 집적시킨 후 신호 경로의 타이밍을 검증하면서 최적화하는 함수들을 목적 함수에 추가시킴으로써 타이밍 최적화 기능을 분석적 초기 배치 단계로 끌어 올리는 시도가 있었다. 이 과정에서 이미 알려진 도착 시간 최소화 함수를 다중 클록을 포함한 설계에도 적용할 수 있도록 개선하였고, 이 때 타이밍 위반이 없는 신호 경로까지 불필요하게 단축될 수 있는 비효율성을 개선하기 위해 최악 음수 슬랙을 최소화하는 함수도 제안하였다. 도착 시간 최소화 함수 적용 시 평균 약 15%의 최악 음수 슬랙 감축 효과를 얻었으며 최악 음수 슬랙 최소화 함수 적용 시 이보다 평균 약 6% 정도 추가 감축 효과도 확인하였다. 이 타이밍 최적화 기능은 성능 및 집적도

향상을 위한 소자들이 상용 FPGA 아키텍처에 포함되면서 발생하는 이슈들을 해결하기 위해 사전 패킹 및 다층 밀도 분석 등의 기법을 적용하여 최근 개발된 분석적 배치 모듈 상에서 추가 기능으로 구현되었다. Xilinx Spartan III XC3S1000 상용 FPGA 마스터와 상용 수준의 설계들 중에서 다양한 프로파일을 포함하도록 선정된 12가지 예제에 적용하여 검증하였다.

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] 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.

[4] 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.

[5] K. Kim, "Evaluation Toolkit for K-FPGA Fabric Architectures," Journal of the IEEK, vol. 49-SD, no. 4, pp. 157-167, April, 2012.

[6] 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.

[7] K. Kim, "Pre-Packing, Early Fixation, and Multi-Layer Density Analysis in Analytic Placement for FPGAs," Journal of the IEEK, vol. 51, no. 10, pp. 96-106, October, 2014.

[8] R. Pattison, Z. Abuowaimer, S. Areibi, G. Gréwal, and A. Vannelli, "GPlace: A Congestion- Aware Placement Tool for UltraScale FPGAs," Proc. of International Conference of Computer Aided Design, November, 2016.

[9] W. Li, S. Dhar, and D.Z. Pan, "UTPlaceF: A Routability-Driven FPGA Placer with Physical and Congestion Aware Packing," Proc. of International Conference of Computer Aided Design, November, 2016.

[10] 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.

[11] Spartan-3 Generation FPGA User Guide, UG331, v1.6, Xilinx Inc., December 3, 2009.

[12] D.J.C. MacKay, "MacOpt-a Nippy Wee Optimizer," http://www.inference.phy.cam.ac.uk/mackay/c/macopt.html, June, 2004.

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

2003년~현재 인천대학교 전자공학과 교수

<주관심분야 : 상위수준합성, 논리합성, 배치, 정적 타이밍 검증, ASIC/FPGA 설계, 설계 데이터베이스, 사용자 인터페이스, 종합설계환경, Reconfigurable Computation, Fault-Tolerance, Embedded Systems, Low-Power Design, Nanoelectronic Architectures, 기계 학습>