Journal of the KNST

2025; Vol.8, No.3; pp.428-431

ISSN: 2635-4926 https://doi.org/10.31818/JKNST.2025.9.8.3.428

Check for updates

Received: 2025/08/06
Revised: 2025/08/24
Accepted: 2025/09/26
Published: 2025/09/30

*Corresponding Author:
Sunghyun Park
Missile System RF & IIR Seeker R&D Lab, LIG Nex1
207, Mabuk-ro, Giheung-gu, Yongin-si, Gyeonggi-do, Republic of Korea
Tel: +82-31-326-9013
E-mail: jhpark80@lignex1.com

Abstract

본 논문에서는 FPGA를 활용하여 가시광센서 기능구현을 위한 logic 타이밍 최적화에 대한 방안을 제시하고자 한다. FPGA는 메모리 접근, 연산 및 센서 제어 등에 필요한 다양한 신호를 생성하고 타이밍에 맞게 연산한다. 이러한 신호 및 연산에는 연산 과정을 통한 지연, 내부 메모리 접근 등 타이밍 지연이 발생하며 이에 시간적 마진 확보가 필요하다. 이에 본 논문에서는 기능 구현을 하면서 발생한 타이밍 문제에 대한 분석과 해결 방안 등 최적화 방안에 대해 소개하고자 한다.

In this paper, we would like to suggest a plan for optimizing Logic timing for implementing visible sensor using FPGA. An FPGA generates various signals required for memory access, computation, and sensor control, and performs computations according to the appropriate timing. Timing delays such as delays through the calculation process and internal memory access occur in these signals and calculations. Therefore, in this paper, we would like to introduce and analyze the timing issues encountered during functionality implementation, as well as discuss optimization methods and potential solutions

Keywords

가시광센서(Visible Sensor),
FPGA(Field Programmable Gate Array),
홀드 시간(Hold Time), 셋업 시간(Setup Time),
시간적 여유(Timing Slack)

Acknowledgement

이 논문은 2025년 정부(방위사업청)의 재원으로 국방과학연구소의 지원을 받아 수행된 연구임.

가시광센서 기능 구현을 위한 FPGA Logic 타이밍 최적화

Optimization of FPGA Logic Timing for Implementation of Visible Sensor

박성현[1*], 유연덕[1], 박진호[2], 김홍락[2]

[1]LIG넥스원 미사일시스템탐색기연구소 선임연구원
[2]LIG넥스원 미사일시스템탐색기연구소 수석연구원

Sunghyun Park[1*], Yeondeok Yoo[1], Jin-Ho Park[2], Hong-Rak Kim[2]

[1]Research engineer, Missile System RF & IIR Seeker R&D Lab, LIG Nex1
[2]Chief research engineer, Missile System RF & IIR Seeker R&D Lab, LIG Nex1

1. 서론

센서 제어 및 영상처리를 위해서는 다양한 칩과 방식을 선택할 수 있으며, 각각의 방식에는 장단점이 존재한다. CPU는 복잡한 연산과 다양한 경우의 수에 유리하며 FPGA는 단순하지만 초기 부팅 속도가 빠르다는 장점이 있다. 본 논문에서는 Xilinx 社의 FPGA 중 Artix Series를 활용하여 가시광센서를 설계 및 구현하였다.

가시광센서 동작을 위해서는 센서 제어, 메모리 접근, 통신, 영상처리 등 다양한 기능이 구현되어야 한다. 하드웨어 언어(HDL) 및 Vivado 툴을 활용해 합성과 implementation을 하게 되고 타이밍에 대한 결과를 확인할 수 있다. FPGA 내부에는 많은 신호들이 여러 모듈의 입력으로 사용되며 그 안에서도 많은 연산과 저장 및 출력 과정이 존재한다. 이러한 과정에서 timing delay가 발생하며 그 신호를 요구하는 입력에 시간 내 도달하지 못하면 timing에 대한 문제가 발생하게 된다. 이를 timing slack(시간적 여유)이라고 하며, 본 논문에서는 기능을 구현하면서 발생한 문제를 소개하고 이를 최적화하는 방식에 대해 소개하고자 한다.

2. CLK 및 setup/hold time

Timing에는 setup time, hold time이 존재하며 양수(positive)의 값이면 회로의 신호가 요구에 만족한다는 의미이다. 반면, 음수

(negative)의 값이면 회로의 신호에 시간적 여유가 없으며 요구되는 setup time/hold time을 만족하지 않는다는 의미이다.

Setup time은 clock의 rising edge가 register에 도달하기 전 확보해야 하는 최소 유효시간이다. Setup time의 요구사항을 만족하지 않는 데이터는 register에 저장되지 않는다.

Hold time은 clock의 rising edge가 register에 입력된 후 데이터가 확보해야 하는 최소 유효시간이다. Hold time의 요구사항을 만족하지 않을 경우 register에 저장되지 않는다.

Setup/hold Time은 비슷해 보이지만 서로 다른 time이다.

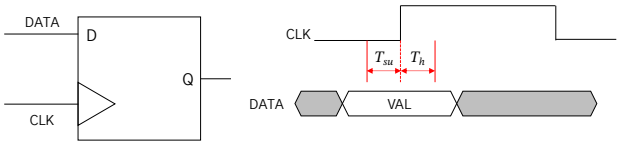

Fig. 1. Setup time and hold time

Clock이 동일하고 register 간 존재하는 time delay는 Fig. 2와 같이 나타낼 수 있으며, Fig. 2에 포함된 parameter 정의는 Table 1과 같다.

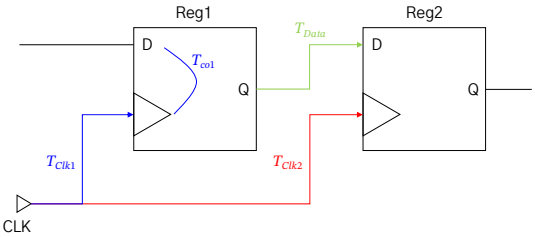

Fig. 2. Inter-register path and timing delay

Table 1. Definition of timing delay

Parameters	Description
T_{clk1}	Net delay to Reg1
T_{co1}	Cell delay to Reg1
T_{clk2}	Net delay to Reg2
T_{data}	Net delay from Reg1 to Reg2
T_{su2}	Setup time for setup check at Reg2
T_{h2}	Hold time for hold check at Reg2

Fig. 3와 같이 요구되는 시간(T_{su2}) 내에 Reg2의 입력에 데이터가 들어오지 못하면 setup time slack (negative)이 발생하며, 이때 Reg1의 출력 데이터는 Reg2에 저장되지 못한다. 이러한 경우 T_{co1}, T_{data} 시간을 단축시켜야 한다.

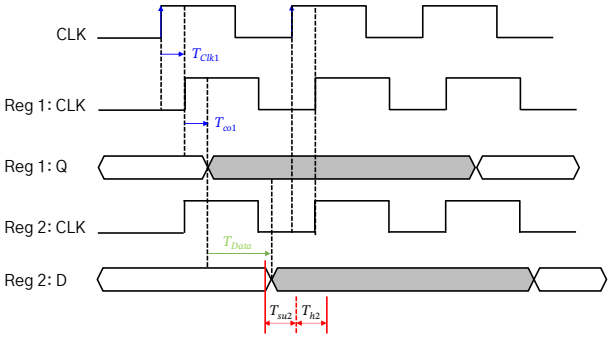

Fig. 3. Setup time slack

Fig. 4는 Reg2의 입력에 데이터가 요구되는 시간(T_{h2}) 이상 유지되지 못하여 hold time slack(negative)이 발생한 경우이다. 이때 Reg1의 출력 데이터는 Reg2에 저장되지 못한다. 이러한 경우 Reg1 출력 데이터를 지연시켜 hold time을 만족시켜야 하지만 setup time 또한 고려하여 지연시켜야 한다.

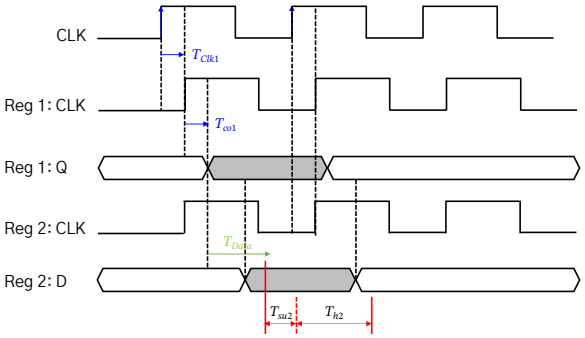

Fig. 4. Hold time slack

3. 가시광센서 구동을 위한 FPGA Logic 모듈

아날로그 방식의 카메라에서 디지털 카메라로 전환되면서 디지털 카메라에 대한 관심이 높다[1]. 최근 디지털 카메라는 센서 자체에서 영상 정보가 디지털 신호로 출력된다. 본 연구에서는 출력된 신호 처리를 위해 Xilinx 社의 Artix Series FPGA를 사용하였다. Fig. 5는 가시광센서의 기능 계통도를 나타낸 것으로, 센서에서 raw 데이터를 받은 후 실시간 불균일 보정 및 데드 픽셀 보정 등 후처리를 통해 LVDS 포맷으로 최종 출력하였다. 붉은색 박스 내 기능 모듈은 FPGA

를 활용하여 설계하였고 그 과정에서 timing slack이 발생하였다.

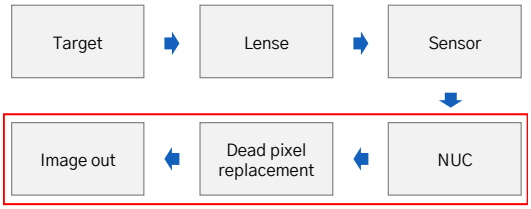

Fig. 5. Functional diagram of the visible sensor

4. Vivado 툴을 활용한 설계 및 timing 검토

Artix는 Vivado 툴을 활용하여 설계하였으며 합성 및 implementation 시 내부 신호들의 timing에 대한 결과를 확인할 수 있다. 가시광센서 기능 구현을 위해서 많은 신호들이 FPGA 모듈 내에서 입력과 출력으로 사용되는데 이 과정에서 timing 문제가 많이 발생한다. 그리고 불균일 보정 모듈에서는 연산을 많이 수행하기 때문에 divider, multiplier 등의 IP가 많이 사용된다. 이러한 IP들은 연산에 많은 시간이 소요되며, divider의 경우 remainder와 같은 결과값이 있으므로 가시광센서 기준 bit에서는 최소한 5 clk latency 이상을 확보해야 한다.

Table 2. FPGA resource utilization for visible sensor (before optimization)

Resource	Utilization (%)
LUT	39.38
LUTRAM	2.56
FF	35.35
BRAM	15.71

Hold time의 경우 모두 만족하여 본 논문에서는 제외하였다. 반면 Table 3는 implementation 후 발생한 setup time slack이다. Worst의 경우 약 4.3 ns (negative)의 slack이 발생하였으며 8,584 point의 신호 중 1,938 point에서 총 3,700 ns 가량의 timing slack이 발생하였다. Register에서 출력된 데이터와 다음 register의 입력 사이에 combination logic이 있으면 delay가 생겨 slack이 발생할 확률이 높아진다. 위와 같이 slack이 발생할 때 합성 및 implementation이 가능한 경우가 있지만 환경시험 혹은 EMC 상황 시 오동작을 야기할 수 있기에 최대한 slack을 없애는 것이 좋다. 모든 point를 논문에 싣기에는 양이 많아 대표 유형과 worst point에 대해 확인하고자 한다. Table 4는 worst setup slack point이다. 보안 상 신호들의 시작-도착 point는 생략하였다.

Table 3. Time result (before optimization)

Category	Valule
Worst negative slack (WNS)	−4.296 ns
Total negative slack (TNS)	−3,692.262 ns
Number of falling endpoints	1,938
Total number of endpoints	8,584

Table 4. Worst setup timing point

Name	Slack	Levels	High fanout	Logic delay	Net delay	Total dealy
Path 61	−4.296	9	2	2.321	2.568	4.889
Path 62	−4.288	9	2	2.313	2.568	4.881
Path 63	−4.212	9	2	2.237	2.568	4.805
Path 64	−4.192	9	2	2.217	2.568	4.785
Path 65	−4.180	8	2	2.204	2.568	4.772
Path 66	−4.172	8	2	2.196	2.568	4.764
Path 67	−4.096	8	2	2.120	2.568	4.688
Path 68	−4.076	8	2	2.100	2.568	4.668

Worst case는 FPGA에 한 모듈에서 나오는 신호가 다른 모듈의 입력으로 들어가는 point에서 setup time slack이 가장 크게 생겼다. 특히 덧셈 연산이 있는 신호의 경우 carry가 발생하기 때문에 이와 같은 경우에서 slack이 많이 발생하였다. 특히 divider IP는 연산 bit가 클수록 IP 내부 latency를 확보했다.

또한 clock이 모듈 내 다른 모듈의 입력 시 내부에서 다시 latch 해주면 안정적으로 설계할 수 있다.

5. Timing slack 최적화 방안

가시광센서를 설계하면서 발생한 slack의 대부분은 inter-clock path에서 발생하였다. Oscillator로부터 FPGA clock 입력을 받고 FPGA는 내부 clock

wizard를 통해 필요한 clock을 생성한다. 이러한 clock은 oscillator의 정수배의 출력 clock으로 생성하는 것이 가장 이상적이나, 그럴 수 없는 경우 실제로 clock이 제대로 생성되는지 확인이 필요하다.

가시광센서를 설계할 때 clock을 latch하는 신호들이 다른 모듈을 지날 때 많은 slack이 발생하였다. 이러한 slack이 발생한 신호에 대해서는 모듈 내에서 clock 혹은 신호를 latch하여 사용하여 setup time을 확보하였다. 그리고 bit가 큰 연산에서는 latency를 확보하였다. 이러한 방법을 통해 timing slack 문제를 해결하였으며 그 이후 합성과 implementation 결과는 Table 5와 같다.

Table 5. Timing result(after optimization)

Category	Valule
Worst negative slack (WNS)	9.428 ns
Total negative slack (TNS)	0.000 ns
Number of falling endpoints	0
Total number of endpoints	28,389

*All user specified timing constraints are met.

기존 worst 경우 약 4.3 ns의 slack이 발생하였으며 8,584 point의 신호 중 1,938 point에서 총 3,700 ns 가량의 timing slack이 발생하였으나 수정 후 약 9.5 ns(positive)를 확보하였다. 또한 1,938 point에서 slack이 발생하였지만 신호의 타이밍을 수정하여 모두 만족시켰다. Table 6은 수정 전 worst setup point이며 현재는 만족하는 것을 확인할 수 있다. 수정 과정에서 약 8,600 point에서 약 28,400으로 늘어났지만 적용된 FPGA로는 충분히 설계가 가능했으며 성능에는 문제 없음을 확인하였다. 수정 후 최종 utilization은 Table 7와 같다.

Table 6. FPGA resource utilization for visible sensor (after optimization)

Resource	Utilization (%)
LUT	42.15
LUTRAM	4.23
FF	38.32
BRAM	20.16

Table 7. Setup timing point(after optimization)

Name	Slack	Levels	High fanout	Logic delay	Net delay	Total dealy
Path 99	18.016	6	6	1.713	4.875	6.588
Path 100	18.044	3	52	0.890	5.077	5.967
Path 101	18.044	3	52	0.890	5.077	5.967
Path 102	18.044	3	52	0.890	5.077	5.967
Path 103	18.044	3	52	0.890	5.077	5.967
Path 104	18.174	1	115	0.890	5.604	6.184
Path 105	18.174	1	115	0.890	5.604	6.184
Path 106	18.189	5	5	1.607	4.818	6.425

6. 결론

본 논문에서는 FPGA를 활용하여 가시광센서를 설계하면서 발생하는 timing slack을 제거하며 logic timing을 최적화하기 위한 방안에 대해 확인하였다. Timing slack은 크게 두 가지 원인 때문에 발생하였다. 첫 번째는 bit 크기에 따른 연산 시간 delay, 두 번째로는 한 신호가 여러 모듈을 거쳐 입력으로 사용될 때 발생하였다. 기능 구현을 위해 timing slack을 고려하지 않은 채 설계하면 slack들이 누적되어 최종 합성 시 문제가 된다. Tool에서는 에러를 띄우나 합성은 할 수 있으므로 일부 설계자들은 이러한 에러를 무시하고 장입하는 경우가 있다. 하지만 이는 최종 성능 및 환경에 따라 장비의 오동작 원인이 된다. 사용하는 FPGA의 스펙이 고사양이라면 slack의 문제를 비교적 쉽게 해결할 수 있다. 하지만 FPGA의 리소스를 최대한 활용하는 경우에는 slacks을 해결할 때 많은 기능 및 Timing이 달라지게 된다. 이에 모듈을 단계적으로 설계할 때 timing slack을 고려하면서 설계하면 최종 단계에서 timing 및 기능을 수정할 일이 적어지며 안전하게 기능 구현을 할 수 있다.

참고문헌

[1] J. Adams, K. Parulski and K. Spaulding, Eastman Kodak Company, “Color Processing in Digtal Cameras,” IEEE MICRO, 1998, pp. 20–30.