The Journal of the Convergence on Culture Technology (JCCT)
Vol. 9, No. 6, pp.967-971, November 30, 2023. pISSN 2384-0358, eISSN 2384-0366

http://dx.doi.org/10.17703/JCCT.2023.9.6.967
JCCT 2023-11-116

ChatGPT을 활용한 디지털회로 설계 능력에 대한 비교 분석

Comparative analysis of the digital circuit designing ability of ChatGPT

남기훈*

Kihun Nam*

요 약 최근에는 다양한 플랫폼 서비스가 인공지능을 활용하여 제공되고 있으며, 그 중 하나로 ChatGPT는 대량의 데이터를 자연어 처리하여 자가 학습 후 답변을 생성하는 역할을 수행하고 있다. ChatGPT는 IT 분야에서 소프트웨어 프로그래밍 분야를 포함하여 다양한 작업을 수행할 수 있는데, 특히 프로그램을 대표하는 C언어를 통해 간단한 프로그램을 생성하고 에러를 수정하는데 도움을 줄 수 있다. 이러한 능력을 토대로 C언어를 기반으로 만들어진 하드웨어 언어인 베릴로그 HDL도 ChatGPT에서 원활한 생성이 예상되지만, 베릴로그 HDL의 합성은 명령문들을 논리회로 구조 형태로 생성하는 것이기에 결과물들의 정상적인 실행 여부를 확인해야 한다. 본 논문에서는 용이한 실험을 위해 규모가 적은 논리회로들을 선택하여 ChatGPT에서 생성된 디지털회로와 인간이 만든 회로들의 결과를 확인하려 한다. 실험 환경은 Xilinx ISE 14.7로 모듈들을 모델링하였으며 xc3s1000 FPGA칩을 사용하여 구현하였다. 구현된 결과물을 FPGA의 사용 면적과 처리 시간을 각각 비교 분석함으로써 ChatGPT의 생성물과 베릴로그 HDL의 생성물의 성능을 비교하였다.

주요어 : 챗GPT, 인간 피드백형 강화 학습, 베릴로그 HDL, 합성, 유한상태머신

Abstract Recently, a variety of AI-based platform services are available, and one of them is ChatGPT that processes a large quantity of data in the natural language and generates an answer after self-learning. ChatGPT can perform various tasks including software programming in the IT sector. Particularly, it may help generate a simple program and correct errors using C Language, which is a major programming language. Accordingly, it is expected that ChatGPT is capable of effectively using Verilog HDL, which is a hardware language created in C Language. Verilog HDL synthesis, however, is to generate imperative sentences in a logical circuit form and thus it needs to be verified whether the products are executed properly. In this paper, we aim to select small-scale logical circuits for ease of experimentation and to verify the results of circuits generated by ChatGPT and human-designed circuits. As to experimental environments, Xilinx ISE 14.7 was used for module modeling, and the xc3s1000 FPGA chip was used for module embodiment. Comparative analysis was performed on the use area and processing time of FPGA to compare the performance of ChatGPT products and Verilog HDL products.

Key Words : ChatGPT(Generative Pre-trained Transformer), RLHF(Reinforcement Learning from Human Feedback), Verilog HDL(Hardware Description Languge), Synthesis, FSM(Finite State Machine)

*정회원, 서경대학교 전자컴퓨터공학과 부교수 (제1저자)
접수일: 2023년 10월 1일, 수정완료일: 2023년 10월 20일
게재확정일: 2023년 11월 5일

Received: October 1, 2023 / Revised: October 20, 2023
Accepted: November 5, 2023
*Corresponding Author: namkh@skuniv.ac.kr
Dept. of Electronics And Computer Engineering, SeoKyeong Univ, Korea

Ⅰ. 서 론

비영리 연구재단 OpenAI가 2018년에 GPT-1 출시 이후 2019년 GPT-2, 2020년 GPT-3, 2023년 GPT-4에 이르기까지 버전이 업데이트 될 때마다 인간이 쓴 글과 구분할 수 없을 정도로 수준 높은 글로 세상을 놀라게 하고 있다. 그로 인해 최근 웹상에서 ChatGPT 관련 논문은 1,370개(2023.1.24. 기준)로 관련 연구가 폭발적으로 급증하는 추세이며, 이미 많은 분야 논문 작성에 ChatGPT가 활용되고 있다. 과학 IT 분야에서도 간단한 프로그램 코드 작성 및 코드상 오류 찾기 및 수정, 프로그램 설치법 안내 등 ChatGPT를 이용해 코드작성, 코드 오류 발견, 코드 수정 등 다양한 프로그램 작업 수행이 가능한 특징을 보이고 있다.[1][2] ChatGPT가 소프트웨어 프로그래밍을 능숙하게 다루듯이 하드웨어 모듈들을 설계할 수 있는지, 설계가 가능하면 기술 수준이 어느 정도인지에 대한 의문이 들어 여러 실험을 통해 설계 능력을 검증하려 한다. 검증하려는 이유는 ChatGPT가 점진적으로 발전한다고 가정한다면 가까운 미래에 하드웨어 엔지니어들의 생존성과도 밀접한 연관이 될 수 있을 것이라 판단되었기 때문이다.

본 논문에서는 ChatGPT을 활용하여 디지털회로 설계 능력을 관찰하고 인간이 설계한 내용을 비교하여 결과의 차이점을 다루려 한다. 2장에서 관련 연구, 3장에서는 연구 방법을 설명, 4장과 5장은 연구 결과 및 결론 순으로 구성된다.

Ⅱ. 관련 연구

ChatGPT(Generative Pre-trained Transformer)를 세부 조정하기 위해 지도 학습과 강화 학습의 조합을 사용하여 만든 RLHF(Reinforcement Learning from Human Feedback) 기술은 훈련 데이터에서 인간의 피드백을 통한 학습으로 거짓이거나 편향된 출력을 최소화 시켜준다.[3] 그림 1의 ChatGPT의 RLHF 단계를 살펴보면 Step1은 사전 교육 작업으로 다수 AI 트레이너가 반복적으로 답변 및 후속 질문을 하여 만들어진 데이터를 병합 후 감독 학습 방식으로 미세조정된 초기 언어 모델이 개발된다. Step2의 보상 모델 준비는 인간의 피드백을 추가하는 단계로서 인간에게 얼마나 잘 인식되는지를 평가하여 모델을 설정한다. Step3 PPO(Proximal Policy Optimization)를 통한 모델 미세조정하여 신뢰성을 최적화하여 인간의 의도에 최대로 부합하는 결과를 도출한다.[4] 하지만 통제 부족, 비교 데이터의 근거 부실, 인간의 다른 선호도 등으로 ChatGPT의 답변이 한계성을 보여준다.

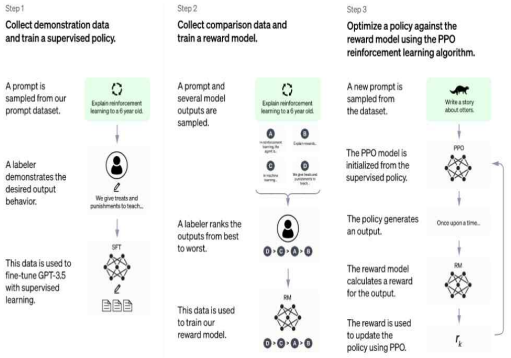

그림 1. ChatGPT의 RLHF 단계
Figure 1. RLHF Steps in ChatGPT

ChatGPT는 프로그래밍에 있어서는 정확한 문제 인식, 명료한 문법, 반복 작업 등의 조건으로 인해 프로그램 코드를 생성한다.[5] 그림 2는 ChatGPT에서 “달력 프로그램 C언어로 작성” 명령으로 만들어진 코드이다.

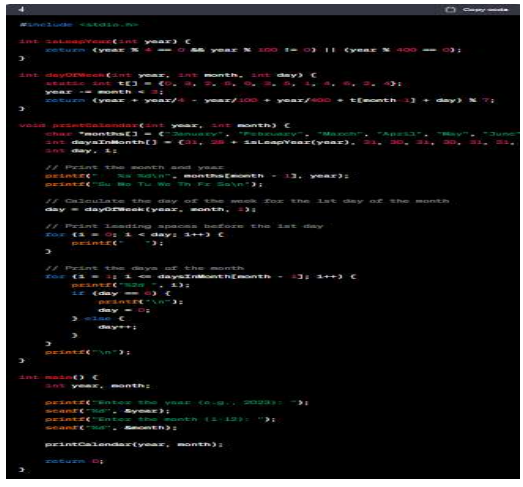

그림 2. ChatGPT가 작성한 달력 C 프로그램 코드
Figure 2. Calendar C Program code written by ChatGPT

프로그램은 간결하고 명확하게 구성되었으며 실행 결과는 그림 3과 같이 실행된다.

```
Enter the year (e.g., 2023): 2023
Enter the month (1-12): 9
   September 2023
Su Mo Tu We Th Fr Sa
                1  2
 3  4  5  6  7  8  9
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
```

그림 3. 달력 프로그램 실행 결과
Figure 3. Calendar program executionresult

Verilog HDL(Hardware Description Languge)은 국제 표준화(IEEE Std. 1364-2005)되었으며, 특정 FPGA 디바이스 및 반도체 제조회사의 공정 기술에 맞도록 회로를 합성(Synthesis)하고 검증이 가능한 언어다.[6] Verilog HDL은 C언어 문법을 기반으로 만들었기에 두 프로그램 형태가 유사한 경우가 많기에 이론상으로는 C언어 경우처럼 Verilog HDL도 ChatGPT로 부터 정확한 결과물을 얻을 수 있을 것으로 판단할 수 있다. 하지만 C언어는 컴파일을 통한 object 파일(기계어)을 생성하여 운영체제에 맞는 링크 작업을 통해 실행되고 Verilog HDL는 코드를 컴파일하여 문법적 부분을 점검하고 난 후 이상이 없으면 합성하여 코드를 gate level netlist로 변환하는 특징을 가진다. 게이트 형식으로 변환된 데이터를 ChatGPT가 어떤 의미로 인식하고 처리할지 실험을 통해 검증하려 한다.

III. 연구 방법

ChaGPT를 활용한 Velilog HDL 코드와 전통적인 방법으로 직접 코드를 생성하여 그 결과들을 비교하려 한다. 우선 16비트 자리올림수 예측 가산기 회로를 구현하기 위해 ChatGPT 4에서 “16-bit Carry-lookahead adder Verliog code”하고 입력하였다. 결과는 그림 4와 같으며, 논리적인 표현에는 문제가 없었으나 합성 결과는 오류가 발생한다. 해당 루프에서 선언된 모듈 인스턴스에 접근을 위한 블록 이름이 선언되어 있지 않기에 단순한 문법 오류가 발생하였기에 다시 ChaGPT에 “generate for (i = 0; i < 16; i = i + 1) begin: propagate와 같이 기술해야 에러가 안 생김” 문장을 입력하여 학습하도록 하였다. 그림 5와 같이 학습 후 사각형으로 표시한 수정된 코드가 각 루프에 적합한 블록 이름들을 생성하여 컴파일 및 합성에 성공하였다.

```verilog
module CarryLookaheadAdder16Bit (
  input [15:0] A, // 16-bit input A
  input [15:0] B, // 16-bit input B
  input Cin,      // Carry-in
  output [15:0] Sum, // 16-bit sum output
  output Cout // Carry-out
);

wire [15:0] G; // Generate signals
wire [15:0] P; // Propagate signals
wire [15:0] C; // Carry signals

// Generate and Propagate signals
genvar i;
generate
  for (i = 0; i < 16; i = i + 1) begin
    assign G[i] = A[i] & B[i];
    assign P[i] = A[i] | B[i];
  end
endgenerate

// Carry Lookahead logic
assign C[0] = G[0] | (P[0] & Cin);

genvar j;
generate
  for (j = 1; j < 16; j = j + 1) begin
```

그림 4. ChatGPT가 생성한 16비트 자리올림수 예측 가산기
Figure 4. 16-bit Carry-lookahead adder generated by ChatGPT

```verilog
module CarryLookaheadAdder16Bit (
  input [15:0] A, // 16-bit input A
  input [15:0] B, // 16-bit input B
  input Cin,      // Carry-in
  output [15:0] Sum, // 16-bit sum output
  output Cout // Carry-out
);

wire [15:0] G; // Generate signals
wire [15:0] P; // Propagate signals
wire [15:0] C; // Carry signals

// Generate and Propagate signals
genvar i;
generate
  for (i = 0; i < 16; i = i + 1) begin : propagate
    assign G[i] = A[i] & B[i];
    assign P[i] = A[i] | B[i];
  end
endgenerate

// Carry Lookahead logic
assign C[0] = G[0] | (P[0] & Cin);

genvar j;
generate
  for (j = 1; j < 16; j = j + 1) begin : carry_logic
    assign C[j] = G[j] | (P[j] & C[j-1]);
```

그림 5. 학습 후 수정된 16비트 자리올림수 예측 가산기
Figure 5. 16-bit Carry-lookahead adder modified after learning

구현된 모듈의 올바른 동작 유무를 검증하기 위한 Test Bench 프로그램도 ChatGPT에 요청해서 얻은 프로그램을 가지고 시뮬레이션한 결과가 그림 6이다.

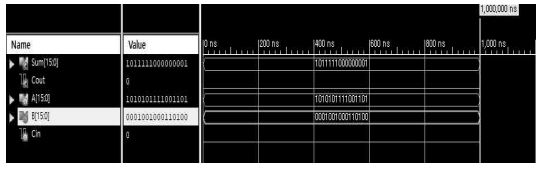

그림 6. 16비트 자리올림수 예측 가산기 시뮬레이션 결과
Figure 6. Simulation results of a 16-bit Carry-lookahead adder

ChatGPT로 의외의 좋은 결과를 얻게 되어 Carry-lookahead adder를 포함한 다른 디지털회로들 Multiplexer, Shift Register, BCD Counter, Ripple Carry Adder, Traffic Signal Controller, Vending

Machine, Booth Multiplier, UART들을 추가하여 ChatGPT로의 설계와 직접 설계한 것을 각각 실행하였다.[7-12] 직접 설계 시 해당 모듈의 최적화 설계를 위해 Date Flow Level, Register Transfer Level(TRL), Behaviral Level를 선택하여 모델링 하였으며 ChatGPT가 설계한 결과들과 1:1 비교 하였다.

실험 환경은 소규모 회로 설계에 적합한 Xilinx Spartan3 XC3S1000 FPGA에서 구현하였고, FPGA에서 사용된 면적(Area)과 속도(Speed)를 측정하였다.[13]

Ⅳ. 연구 결과

설계 후 합성 결과는 표 1과 같다. ChatGPT는 Multiplexer, Shift Register, BCD Counter, Ripple Carry Adder같은 단순 회로들은 합성되어 정상적인 동작을 확인하였다. 그러나 유한 상태 기계(FSM) 기법이 적용되거나 알고리즘이 포함된 중급 모듈에 한에서는 기본적인 형태만을 제시하고 추가적인 요소를 인간이 직접 수정할 것을 권유하였다. ChatGPT 제시한 기본형태를 바꾸지 않은 선에서 코드를 수정하여도 합성이 되질 않기에 유한 상태 기계의 이해와 분석까지는 학습이 부족한 것으로 보인다.

표 1. 모듈별 합성 결과
Table 1. Synthesis result by module

Module	Human Engineer	ChatGPT	ChatGPT 수정 후
4bit 16x1 Multiplexer	Successfully (정상 작동)	Successfully (정상 작동)	-
16bit SIPO Shift Register	Successfully (정상 작동)	Successfully (정상 작동)	-
Binary coded Decimal counter 4bit	Successfully (정상 작동)	Successfully (정상 작동)	-
Ripple Carry Adder 16bit	Successfully (정상 작동)	Failed (일부 수정)	Successfully (정상 작동)
Carry Lookahed Adder 16bit	Successfully (정상 작동)	Failed (일부 수정)	Successfully (정상 작동)
(FSM) Traffic Signal Controller	Successfully (정상 작동)	Failed (전체 수정)	Successfully (오작동)
(FSM) Vending Machine	Successfully (정상 작동)	Failed (전체 수정)	Successfully (오작동)
Booth Multiplier 8bit	Successfully (정상 작동)	Failed (전체 수정)	Successfully (오작동)
UART(RX/TX)	Successfully (정상 작동)	Failed (전체 수정)	Successfully (오작동)

표 2에서는 FPGA에서 실행 시 사용되는 면적과 속도를 측정한 결과를 간단히 비교할 수 있게 구성하였다.

표 2. 모듈 성능의 비교
Table 2. Comparison of Module Performance

Bit	Module	Xilinx Spartan3 XC3S1000			
		Human Engineer		ChatGPT	
		Speed (ns)	Area (LUTs)	Speed (ns)	Area (LUTs)
4	16x1 Multiplexer	5.41	36	5.87	33
16	SIPO Shift Register	7.24	45	7.24	45
4	Binary coded Decimal Counter	4.45	24	3.14	23
16	Ripple Carry Adder	5.23	100	8.97	101
16	Carry Look ahed Adder	5.31	143	6.34	131
-	FSM Traffic Signal Con.	4.21	152	-	12
-	FSM Vending Machine	2.40	66	-	12
8	Booth Multiplier	17.41	173	-	12
8	UART(RX/TX)	6.56/ 5.07	123/ 114	-	12/ 12

Ⅴ. 결 론

인간이 설계한 디지털회로와 ChatGPT가 설계한 디지털회로의 성능 결과를 비교하였다. 예상 밖으로 ChatGPT가 소규모 디지털회로 설계에서는 탁월한 능력을 보였다. 이를 잘만 활용한다면 설계 시간 및 비용을 절약하는 효과가 있을 것으로 보인다. 하지만 특정 알고리즘을 적용하거나 다소 복잡한 구조의 회로 설계에 있어서는 기술적인 한계를 드러내어 인간 엔지니어의 도움이 절대적으로 필요해 보인다. 하지만 가까운 미래에 ChatGPT가 지속적인 학습을 통해 발전하여 현재의 문제점들을 극복한다면 엔지니어들의 고용시장에

서 악영향을 미치는 부정적인 효과가 나타날 수 있다는 우려가 든다. 부정적인 문제가 현실화되기 전에 미리 대응 방안도 차차 모색해야 할 것으로 생각한다. 현재 특수 목적의 컨트롤러를 설계 중인데 자주 ChatGPT에 접속하여 발전과정을 지속적으로 관찰하려 한다.

References

[1] OpenAI, "GPT-4 Technical report", https://arxiv.Iorg/abs/2303.08774, 2023, DOI: 10.48550/arXiv.2303.08774
[2] Peng, K., Ding, L., & Tao, D, "Towards making the most of ChatGPT for machine translation", https://arXiv.org/abs/2303.13780, 202 3, DOI: 10.48550/arXiv.2303.13780
[3] https://openai.com/blog/chatgpt
[4] John Schulman, Filip Wolski, Prafulla Dhariwal, ALec Radford, Loeg Klimov, "Proximal Policy Optimization Algorithms", Machine Learnig computer Science, 2017, DOI: 10.48550/1707.06347
[5] Seul Ki Kim, "Exploring the Possibility of Using Generative Artificial Intelligence for Programming Education: Focusing on ChatGPT", The Journal of Koran associatioon of computer education, Vol. 27, No. 2, pp.151-154, 2023.
[6] Johny Srouuji, Tom Fitzpatrick, Neil Korpusik, "IEEE Standard for Verilog Hardware Description Language", IEEE Computer Society, 2006.
[7] Shanzhen Xing, W.W.H. Yu, "FPGA Adder: performance evaluation and optimal design", IEEE Design & Test of Computers, Vol. 15, pp.24-29, 1998, DOI: 10.1109/54.655179
[8] P. Metzgen, D. Nancekievill "Multiplexer restructuring for FPGA implementation cost reduction", Proceedings of the 42nd annual Design Automation Conference, pp.421-426, 2005, DOI: 10.1145/1065579.1065692
[9] K. Swetha Reddy, B.B. Shabarinath, "Timing and Synchronization for Explicit FSM Based Traffic Light Controller", IEEE 7th International Advace Compution Conference(IACC), 2017, DOI: 10.1109/IACC.2017.0114
[10]M Irmansya, E Madona, "Newspaper Vending Machine", WMA-2, 2018, DOI: 10.4108/eai.24-1-2018.2292389
[11]Sungyeong Jang, Sangwoo Park, Guyun Kwon, Taeweon Suh, "Design and Evaluation of 32-Bit RISC-V Processor Using FPGA", Vol. 11, No. 1, pp.1-8, 2022, DOI: 10.3745/KTCCS.2022.11.1.1
[12]Kihun Nam, "A design of the ALU for Convolution Neual Network of operation processing", Aisa-pacific Journal of multimedia Services Convergent with art, Humanities, and Sociology, Vol. 7, No.11, pp.879-886, 2017, DOI: 10.14257/ajmahs.2017.11.64
[13]SangWook Lee, Bosen Lee, Taeweon Suh, "FPGA Performance Evalution According to HDL Coding Style", Proceedings of the Korea Information Processing Society Conference, Vol. 18, No. 5, pp.62-65, 2011, DOI: 10.3745/PKIPS.y2011m11a.62

※ This work was supported by Seokyeong University in 2022.