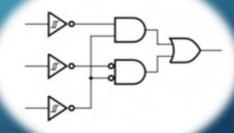


SYMULATOR UKŁADÓW CYFROWYCH

BARTŁOMIEJ KRASOŃ

ISTOTA PROGRAMU

- skierowany na ideę edukacyjną
- program konsolowy (biblioteka klas), z opcją rozbudowy
- interfejs uproszczony (tworzony celem reprezentatywnym i testowym)
- odzwierciedlenie rzeczywistego problemu
- narzędzie pomocne przy modelowaniu układów cyfrowych



INTERFEJS PROGRAMU

Program wykonuje się w głównej pętli akcji, która dostarcza następujące operacje:

- dodawania elementu logicznego
- dodawania połączeń między elementami
- czyszczenia układu
- podglądania stanów logicznych na elementach całego układu

DIAGRAM KLAS LogicElement Wire - index: int - value: int - inElem: LogicElement* + connectList: vector<Wire> Constant - outElem: LogicElement* value: int Extends + RefreshLogic() + set/N(i: int) + selectInput(): int + getOut(o: int) + selectOutput(): int + getOut(o: int) + Update() + RefreshLogic() + getSuccessor(): LogicElement* Extends Extends Extends Extends MUX Gate Latch DMUX - input1: int - inputS: int type: mxType (enum) - type: mxType (enum) - input2: int - inputR: int - inAddress: vector<int> - inAddress: vector<int> - output: int - outputQ: int - inInform: vector<int> - input: int - outputNotQ: int - output: vector<int> - output: int + setIN(i: int) + getOut(o :int) + setIN(i: int) + setIN(i: int) + setIN(i: int) + getOut(o: int) + getOut(o: int) + getOut(o: int) + RefreshLogic() + RefreshLogic() + RefreshLogic() AND OR XOR + RefreshLogic() + RefreshLogic() + RefreshLogic() NAND NOR + RefreshLogic() + RefreshLogic()

STRUKTURA DANYCH

MAPA: map<int, LogicElement*> listOfLogicElements

- unikalność kluczy, wykorzystujących indeksy elementów logicznych

WYKORZYSTYWANE TECHNIKI

- Wyjątki (z własnymi klasami wyjątków)
- Kontenery STL (główny kontener mapy, jak i pomocnicze vectory np. połączenia każdego elementu)
- Iteratory i algorytmy STL
- RTTI
- Wątki (?)