Szymon Wiszczuk 241343

Krystyna Łosieczka 241328

**Architektura Komputerów 2**

Implementacja podstawowych funkcji obliczeniowych zmiennoprzecinkowych poprzez implementację FPU operacjami stałoprzecinkowymi przy pomocy Arduino Uno.

Prowadzący: dr.inż. Dominik Żelazny



(Kod programu dostępny pod linkiem: <https://github.com/golota60/IEEE-754-Calculator-Arduino>)

1. **Skład Hardware’owy projektu**

Kalkulator składa się z następujących części, odpowiednio z cenami:

-Arduino Uno(Klon) - ~20zł

-Keypad Matrix 4x4 - ~4zł

-LCD 16x2 z modułem I2C ~8 zł

-Kable M-F oraz F-F,przycisk(reset) ~2zł

-Battery Pack - ~6zł

-Mini Breadboard – ~4zł

-Baterie AA x6 ~3zł

-Kartonowe Opakowanie – Bezcenne

Całkowity Koszt Projektu: ~47zł

1. **Przebieg Projektu**

Naszym celem przy tworzeniu projektu było stworzenie kalkulatora przy pomocy Arduino, wyświetlacza oraz klawiatury. Pierwszym napotkanym problemem było zadecydowanie czy chcemy stworzyć kalkulator liczb zmiennoprzecinkowych czy stałoprzecinkowych. Różnica jest znacząca: Arduino wyposażone jest w procesor ATMega328, który nie posiada FPU, czyli dodatkowych instrukcji procesora pozwalających na natywne obliczanie liczb zmiennoprzecinkowych.

Zdecydowaliśmy się na stworzenie kalkulatora liczb zmiennoprzecinkowych. Początkowo chcieliśmy użyć łączonego kodu C z Assembly, jednak po implementacji stałoprzecinkowych operacji Dodawania i Odejmowania, zdaliśmy sobie sprawę, że Assembler ATMegi nie jest zbyt przyjazny dla użytkownika, a w dokumentacji procesora trudno jest się odnaleźć. Nie mogliśmy znaleźć żadnej książki ani osoby, która miała doświadczenie z tego typu Assembly, więc porzuciliśmy ten pomysłu.

W związku z tym postanowiliśmy skorzystać z mikrokontrolera nieco bardziej zaawansowanego, mianowicie STM32F334-Nucleo z procesorem ARM-Cortex M4 na pokładzie, który posiada już moduł FPU. Niestety po paru tygodniach prób przeportowania obecnego projektu na tę płytkę poddaliśmy się, ponieważ sposób programowania tej płytki bardzo różni się od Arduino, a biblioteki Open-Sourcowe Stm32duino, zbliżające sposób programowania jednego mikroprocesora do drugiego niestety nie obsługiwały w pełni wszystkich możliwości Arduino. Doprowadziło to do wielu problemów – tak jak na przykład nieprawidłowe działanie naszego LCD.

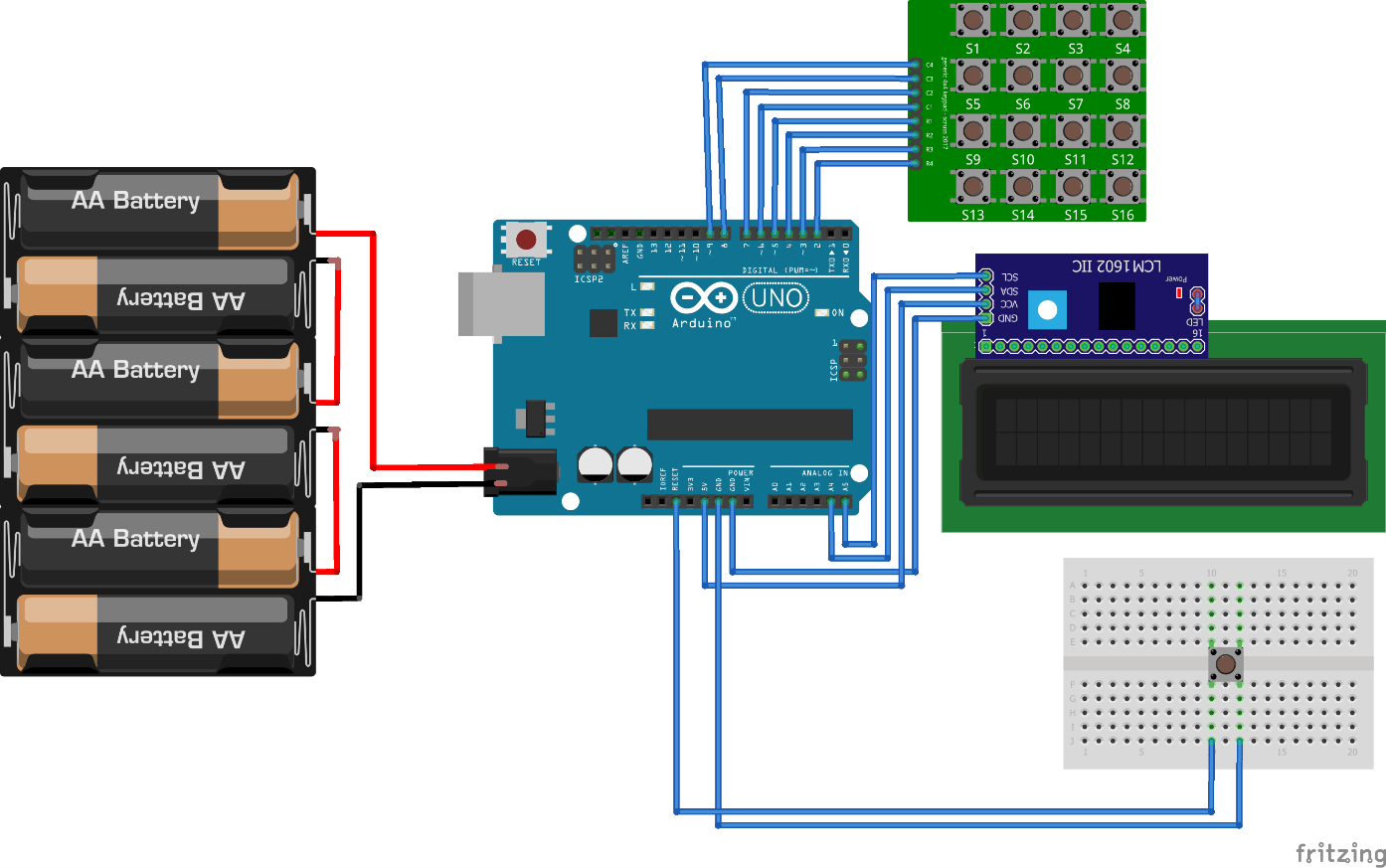
Powróciliśmy do Arduino, jednak biorąc pod uwagę nieczytelność assemblera ATmegi328, postanowiliśmy zaimplementować instrukcje FPU w języku C. Aby to zrobić, inspirowaliśmy się tym jak procesory bez FPU faktycznie działają na liczbach zmiennoprzecinkowych. Po zwróceniu się ku literaturze, dowiedzieliśmy się, że liczby te są zmieniane na liczby w standardzie IEEE 754, a następnie są wykonywane na nich operacje stałoprzecinkowe. Wiemy jak działa standard IEEE 754 z kursu Architektura Komputerów I, więc ten sposób wykonania kalkulatora został naszym ostatecznym wyborem.

W międzyczasie tych wszystkich zawirowań naszej implementacji kalkulatora, nie próżnowaliśmy. Wszystkie komponenty połączyliśmy razem i kupiliśmy battery packa który mieści 6 baterii, ponieważ zależało nam na tym, aby nasz kalkulator mógł działać bez podłączenia do komputera. Zaprogramowaliśmy również nasze LCD i klawiaturę oraz całe GUI tak, aby jedyne co zostało nam do zrobienia to były operacje obliczeniowe. Wykonaliśmy również kartonowe opakowanie, aby ukryć mikroprocesor, baterie i większość kabli, żeby użytkownik widział tylko ekran LCD oraz klawiaturę.

W związku z ilością napotkanych problemów i trudnością projektu, nie udało nam się zaimplementować dzielenia zmiennoprzecinkowego przez brak czasu, co jest szkodą, ponieważ dopełniałby on zestaw wszystkich podstawowych funkcji obliczeniowych.

1. **Ostateczna postać projektu**

**3.1. Połączenie Hardware’u**



**3.2. Opis Operacji Obliczeniowych**

Największą jednostką zmiennoprzecinkową Arduino Uno jest float. Przez to, że ma on rejestry jedynie 8-bitowe, a float ma 4 bajty, wiemy, że float jest wielkości 32-bitowej i operacje na nim wykonywane są pojedynczej precyzji. Liczby w formacie Znaku, Mantysy i Eksponenty uzyskujemy poprzez łączenie instrukcją union floata oraz dwóch int-ów i jednego long-a. Dwa inty dla Eksponenty oraz Znaku i jeden Long dla Mantysy, ponieważ mantysa może przyjmować tak wielkie wartości, że 16-bitowy int Arduino bywa zbyt mały.

Całość obliczeń uzyskiwania jest na podstawie operacji stałoprzecinkowych oraz manipulacji bitów, czyli ich przesuwaniu. Używając tych narzędzi tworzone są algorytmy dodawania, odejmowania oraz mnożenia. Stworzyliśmy je posiłkując się wiedzą nabytą na Architekturze Komputerów I, i to właśnie algorytmy na tym przedmiocie poznane implementowaliśmy w programie. Dokładne operacje są wytłumaczone w kodzie programu, jednak w związku z naturą projektu i tak wymagają rozpisania, aby w pełni go zrozumieć.

1. **Wnioski**

Projekt był jednym z bardziej wymagających jakie do tej pory przyszło nam realizować w ramach nauki na uczelni. Jednak mimo to mieliśmy okazję zobaczyć i przetestować wiedzę nabytą w pierwszej części kursu w praktyce. Możliwość użycia wiedzy hobbystycznej, takiej jak nasze zainteresowanie Arduino i poznanie jak działa ono na najniższym poziomie, pozwoliło nam zrozumieć procesory oraz ich architektury.

Przez naturę projektu i problemy opisane w przebiegu, nauczyliśmy się więcej niż na jakimkolwiek innym projekcie. Napotykaliśmy wiele problemów i musieliśmy posiłkować się literaturą oraz dokumentacją, aby dowiedzieć się w jaki sposób je rozwiązać. Poznaliśmy jak działają mikroprocesory i nauczyliśmy się wielu środowisk w jakich się je programuje. Jeżeli chodzi o kod, nasze umiejętności programowania zostały poddane testowi. Nauczyliśmy się również wielu funkcjonalności języka C. W celu wykonania dokumentacji zaznajomiliśmy się także z tym, jak tworzyć profesjonalne schematy w zakresie elektroniki.