<table>
<tr><td>  </td><td>Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie.
Wydział Informatyki, Elektroniki i Telekomunikacji</td></tr>
<tr><td colspan="2">Sensory w aplikacjach wbudowanych
SPRAWOZDANIE
Rok I, Systemy wbudowane, Elektronika i Telekomunikacja IIst.</td></tr>
<tr><td colspan="2">Temat: Moduł z sensorami oparty na procesorze STM32F103, zgodny z wyprowadzeniami Mikrobus™</td></tr>
<tr><td>Zespół:
1. Mateusz Kozyra
2. Mirosław Wiącek
3. Radosław Sajdak</td><td>Oceny indywidualne:
1.
2.
3.</td></tr>
<tr><td>Data oddania sprawozdania:</td><td>Ocena sprawozdania:</td></tr>
<tr><td>Uwagi prowadzącego zajęcia:</td><td>Informacje dodatkowe:</td></tr>
</table>

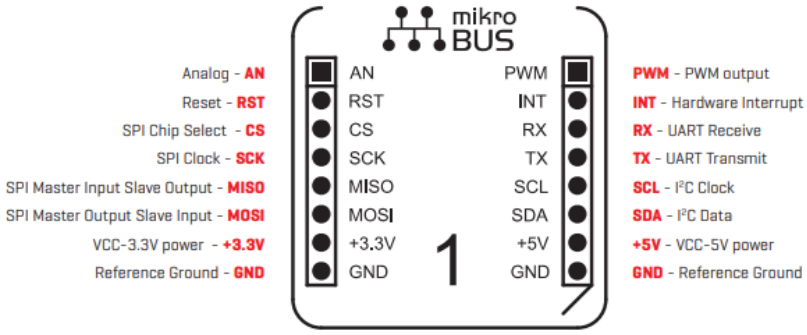
Spis treści

1. Wstęp

Założeniem projektu, było stworzenie modułu sensora, zgodnego ze standardem mikroBUS™. Zespół, korzystając z okazji na zdobycie nowych umiejętności, zdecydował się rozszerzyć założenia projektowe o dwa dodatkowe sensory (co daje łącznie trzy układy pomiarowe na jednej płytce). Układy te, komunikują się z mikroprocesorem STM32F103 znajdującym się na tej samej płyce. W procesorze, dane są przetwarzane, nakładając dodatkową warstwę abstrakcji dla obsługi sensorów. Z płytką, można komunikować się przy użyciu interfejsu UART. Dodatkowym założeniem projektu, było różne podejście do tworzenia oprogramowania, każdego z członków zespołu. Zdecydowano, że jeden będzie tworzyć kod, testując go z mikroprocesorem STM32F4, kolejny bezpośrednio na wykonanej płyce, a ostatni, bez bezpośredniej styczności ze sprzętem. Podejście to, pozwoliło spojrzeć na pojekt z innej perspektywy, jak gdyby zespół był większy i rozproszony. Całe proces tworzenia był kontrolowany przy użyciu platformy Github, pozwalającej na wygodną kontrolę wersji oraz dyskusję nad kolejnymi zmianami w kodzie.
W niniejszym sprawozdaniu, skupiono się przede wszystkim na procesie tworzenia modułu oraz popełnionych błędach, nie pomijając krótkiego wstępu teoretycznego. Całość, zakończona jest wnioskami na temat procesu oraz zdobytych umiejętności.

1.1. mikroBUS™

MikroBus™jest standardem definiującym układ gniazd oraz wyprowadzeń na płytce, zawierającej układ scalony (np. sensor). Standard opisuje również dokładne wymiary płytki, a także narzuca warstwę nadruków na płytce. Założeniem mikroBUS™, jest umożliwienie szybkiego i łatwego rozbudowywania układów na rzecz prototypowania. Dostarczane przez wielu producentów płytki z gotowymi gniazdami, pozwalają w prosty sposób uruchomić dowolny sensor, a następnie stworzyć dla niego oprogramowanie. Tym samym, mikroBUS™okazuje się świetną alternatywą dla niestabilnych płytek stykowych. Na rysunku 1.1 przedstawiono zdefiniowany w standardzie pinout.
Ze względu na zmienione założenia projektu, stworzona płytka zgadza się ze standardem co do wyprowadzeń i wymiarów, jednak nie posiada naniesionych nadruków. Wynika to wprost z braku miejsca, na płytce. Dodatkowo, kilka sensorów na jednej płytce, również nie jest zgodne ze standardem.



Rys. 1.1. Specyfikacja wyprowadzeń mikroBUS™[1]

1.2. STM32F103

Do stworzenia projektu, wybrano procesor STM32F103C8 z rdzeniem ARM®Cortex®-M3. Wybór, podyktowany był przede wszystkim dostępnością, ponieważ układ ten, w obudowach LQFP48 znajdował się w prywatnych zasobach zespołu. Obecnie, na rynku dostępne jest niewiele procesorów, a części elektroniczne są drogie. Z tego powodu, zdecydowano się korzystać z tego, co było dostępne. Nie oznacza to jednak, że procesor ten był pod jakimkolwiek względem ograniczający. Najistotniejszymi z jego cech są:

- 2 interfejsy I^2C
- 3 interfejsy USART
- 2 interfejsy SPI
- Zasilanie 2.0 - 3.6V
- Zewnętrzny zegar 4 - 16MHz
- 12-bitowe, wielokanałowe przetworniki A/D

Układ, programowany jest przy użyciu interfejsu SWD oraz programatora ST-link. Fakt ten, znacząco ułatwił wykonanie działającego układu. Ze względu na przyjęte założenia i wybrane sensory, wybrany procesor bardzo dobrze wpasowywał się w projekt. Dla wykonania oprogramowania, istotne były również dostarczone przez ST biblioteki oraz oprogramowanie STM32CubeMX, wprowadzające wysoki poziom abstrakcji w tworzeniu oprogramowania. Pozwoliło to uniknąć często czasochłonnego, ręcznego pisania do rejestrów procesora, w celu konfiguracji peryferiów. Były również momenty, w których HAL okazał się problemem, co wspomniane zostało w dalszej części sprawozdania.

1.3. STS30

Krótki opis sensora Mirka

1.4. BMP280

Krótki opis sensora Matiego

1.5. MQ2

Krótki opis sensora Matiego

1.6. Makefile

Jak wspomniano we wstępie, jednym z założeń, było tworzenie oprogramowania równolegle, na dwóch różnych procesorach. STM32F1 i STM32F4. Procesory te, należąc do zupełnie różnych rodzin, korzystają z innych bibliotek, a do ich konfiguracji niejednokrotnie wymagane są różne kroki. Z tego powodu, w projekcie wykorzystano program *Make*. Pozwala on, na podstawie pliku konfiguracyjnego *makefile*, zdefiniować sposób kompilacji oraz ścieżki bibliotek wykorzystywanych przez kod. Narzędzie to, jest stosunkowo proste w konfiguracji, a jej przykłady są ogólnodostępne w internecie. Dodatkowym argumentem przemawiającym za jego wykorzystaniem, była chęć jego przetestowania w praktyce.

1.7. Github

Github, to serwis internetowy pozwalający przechowywać projekty z wykorzystaniem systemu kontroli wersji. Korzysta on z systemu Git opublikowanego na licencji GNU GPL 2 (wolne i otwarte oprogramowanie). System ten, jest obecnie wykorzystywany przez wielu programistów, a jego znajomość często wymagana jest przez pracodawców. Github, poza kontrolą wersji, dostarcza wygodne i przejrzyste środowisko w przeglądarce, pozwalające programistom na np. przegląd kodu współpracowników. Fakt ten, został wykorzystany podczas tworzenia projektu. Pozwoliło to nie tylko na poprawienie jakości tworzonego oprogramowania, ale również skonfrontowanie różnych podejść do współpracy nad kodem.

2. Wykonanie projektu

2.1. Projekt płytki

2.1.1. Schemat

Tutaj wkleję schemat, opiszę krótko jak zdecydowałem się go podzielić. Warto wspomnieć o tym, że korzystałem przy projektowaniu z dokumentacji producentów, a i tak się pomyliłem. Trzeba też koniecznie powiedzieć, że komponenty dobierałem samodzielnie na podstawie dostępności na Mouserze i dokumentacji. Koniecznie napisać, o konwerterze do debugu kodu.

2.1.2. Layout

Pokazać warstwy elektryczne. Zdjęcia wydrukowanych płytek. Jakie problemy się pojawiły (1 raz z Kicadem, więc np. okazało się, że domyślnie ścieżki ma całkiem szerokie. Podobnie viasy). Brak miejsca na wyprowadzenia SWD

2.1.3. Popełnione błędy oraz wykonane przeróbki

- Pokazać odcięcie LDO konwertera
- Bramka wisząca w powietrzu
- Zła przetwornica 5V (Vin > Vout)
- Źle wsadzony mosfet. Pokazać możliwe przeróbki (OPAMP, rezystory, zdjęcia z prób i cięć).
- Bypass przetwornicy
- Ręczne lutowanie SWD do procka

2.2. Oprogramowanie

2.2.1. Biblioteki peryferiów

Krótko opisać jakie były założenia przy tworzeniu peryferiów (rozbicie drzewka kodu dla porządku, ustandaryzowane kody błędów, osobne libki na wszystko itp.)

2.2.2. API

Założenia API. Jak działa, wrzucić grafy, listę komend(?), mechanizmy rejestracji komend jako coś, co pozwala łatwo rozbudować soft do innej aplikacji.

2.2.3. Niewykorzystane możliwości

Nie wiem, tutaj mógłbym może napisać że zrobiłem kozak timery, ale w sumie to nie wiadomo po co bo API jest bezobsługowe xD No i może o tym, że Mati nie wiadomo po co pisał na F4...

2.2.4. System kontroli wersji

IMHO spoko też napisać, o tym, że korzystaliśmy z tego gita mocno, dbaliśmy o porządek, robiliśmy review itp. To jednak zjadło kupę czasu.

3. Prezentacja działania

Wrzucić tutaj fotki z cutecoma i pokazać, że działa. Może jakieś zdjęcia że przed initem lampka nie świeci, a po świeci xD

4. Wnioski

No na pewno napisać, że inaczej bym PCBka zrobił. Że dużo było czytania dokumentacji, trzeba było sobie przypomnieć sporo z elektroniki, a i tak pojawiły się błędy. Na pewno, że 0805 są za duże. Że robienie review pomaga obu stronom czegoś się nauczyć. Że sensory na PCB muszą być przemyślane (MQ + temperatury). Że mimo tego, że N osób widziało schematy i layout, to błędy przeszły produkcję i dużo czasu straciło się na debug zasilaczami, oscyloskopami itp.. Ale trzeba też powiedzieć, że robiliśmy projekt 4fun i dużo się nauczyliśmy.

Bibliografia

[1] MikroElektronika. *MikroBUS standard specification*.