Projekt z przedmiotu
“Programowanie niskopoziomowe”

Temat: bootloader

Autor: Dominik Kobyra

1. Opis tematu

1.1 Ogólnie o bootloaderze

Bootloader, czyli program rozruchowy, to program napisany w celu załadowania systemu. Zajmuje dokładnie 512 bajtów, przy czym na dwóch ostatnich bajtach znajduje się specjalny znacznik pozwalający na wykrycie programu rozruchowego przez BIOS, sekwencja 0x55 i 0xAA. Po znalezieniu takiego znacznika, program jest ładowany do pamięci pod adresem 0x0000:0x7C00.

1.2 Tryb rzeczywisty

Podczas ładowania programu rozruchowego, procesor działa w trybie rzeczywistym. Przekłada się to na nielimitowany, bezpośredni dostęp do całej adresowalnej pamięci (przy czym wielkość tej pamięci to 1 MiB), dostęp do adresów wejścia/wyjścia, a także do komponentów peryferyjnych. Tryb ten nie oferuje jednak żadnego rodzaju ochrony pamięci, wielozadaniowości ani dostępu do informacji o poziomach zaufania (privilege levels).

1.3 Tryb chroniony

Współczesne procesory intelowskie działają na chronionym trybie operacyjnym. Pozwala on na pracę z wirtualnymi przestrzeniami adresowymi, adresowalnymi do 4 GB pamięci, odblokowuje również możliwość stosowania ścisłej ochrony pamięci przez system i ochrony sprzętowych operacji wejścia/wyjścia.

1.4 Przerwania BIOSu

Przerwania BIOSu dostarczają szeregu użytecznych funkcjonalności, używanych szczególnie podczas rozruchu komputera. Dostęp do funkcji BIOSu jest najprostszy z poziomu rzeczywistego trybu pracy procesora, wystarczy użyć odpowiednich instrukcji w kodzie asemblera.

W celu skorzystania z określonej funkcji BIOSu, zwykle należy ustawić rejestr AH na konkretną wartość oraz wykonać instrukcję `INT OPCODE`, przy czym opcode to numer wybranego przerwania. Wartość z rejestru AH połączona z numerem przerwania powoduje wywołanie pożądanej funkcji. Ponadto, do innych rejestrów procesora można wprowadzić argumenty, które zostaną użyte przez funkcję.

Ważne: należy pamiętać, że zarówno wartość podawana do instrukcji INT, jak i do rejestru AH, to wartości szesnastkowe.

Jeżeli procesor działa w trybie chronionym, praktycznie wszystkie funkcje BIOSu stają się dla niego niedostępne. Jednym ze sposobów skorzystania z funkcji BIOSu podczas pracy w tym trybie jest chwilowe przejście do trybu rzeczywistego, skorzystanie z funkcji, oraz powrót do trybu chronionego. Metoda ta nie jest jednak pozbawiona wad, dlatego wszelkie operacje wymagające przerwań BIOSu powinny zostać wykonane zanim procesor przejdzie w tryb chroniony.

Poszukując funkcji BIOSu potrzebnej do wykonania określonego zadania, warto posiłkować się tego rodzaju listą https://en.wikipedia.org/wiki/BIOS_interrupt_call#Interrupt_table. W kolumnie "Interrupt vector" podane są wartości, które należy przekazać do instrukcji INT, natomiast w drugiej kolumnie opisane są funkcje dostępne dla określonego przerwania wraz z wartością, którą należy umieścić w rejestrze AH.

Przykładowo, dla "Interrupt vector" równego 10h, dostępna jest lista operacji wykorzystujących kartę graficzną. Widzimy, że dla AH równego 0Eh, wywołana zostanie funkcja wypisująca znak na ekran, ale ilość podanych informacji jest niewystarczająca do prawidłowego skorzystania z tej funkcji. Należy zatem przeczytać dokładniejszy opis przerwania INT 10h, na przykład tutaj https://en.wikipedia.org/wiki/INT_10H. W sekcji "List of supported functions" widzimy szczegółowy opis każdej funkcji wraz z dodatkowymi parametrami oraz zwracanymi wartościami. W kolumnie "Parameters" widzimy listę argumentów przyjmowanych przez funkcję AH=0Eh, czyli AL, BH i BL. Rejestru AL używamy do przechowania wypisywanego znaku, rejestry BH i BL to kolejno numer strony i kolor znaku.

Analogicznie wygląda to w przypadku korzystania z innych przerwań. Przykładowo, dla AH=03h (czyli pobranie pozycji i kształtu kursora na ekranie) musimy podać tylko numer strony w rejestrze BH i wywołać INT 10h, wartości wynikowe zostaną zwrócone do rejestrów AX, CH, CL, DH, DL.

2. Opis zadań

Na zajęciach laboratoryjnych studenci będą zajmować się pisaniem prostych skryptów funkcjonujących jako programy rozruchowe. Treści zadań są zgodne z wcześniej ustalonym planem, z którym studenci mieli okazję się zapoznać. Do zrealizowania przygotowałem pierwsze trzy zadania wymienione w planie, dla każdego z nich został utworzony plik .asm, który należy uzupełnić.

Dodatkowo, studentom został udostępniony skrypt run.sh, który można wykorzystać do uruchamiania tworzonych programów. W tym celu należy podać do skryptu nazwę pliku, który chcemy uruchomić, przykładowo:

```
./run.sh zadanie_1.asm
```

Powyższa komenda dokona asemblacji kodu, utworzy plik zadanie_1.bin, a następnie plik ten zostanie uruchomiony przez emulator QEMU.

Uwaga: w celu opuszczenia okna emulatora, należy nacisnąć **ESC + 2**, wpisać **q** oraz potwierdzić klawiszem **Enter**. Skrót ESC + 2 przełącza nas do konsoli QEMU, instrukcją q wychodzimy z emulatora.

Zadanie 1

Celem zadania pierwszego jest zapoznanie studentów z zachowaniem emulatora QEMU w przypadku braku wykrycia urządzenia bootowalnego, a także jego zachowaniem, gdy dostarczony skrypt zostanie poprawiony i wykonany.

Treść zadania:

Proszę uruchomić plik zadanie_1.asm przy pomocy dostarczonego skryptu run.sh oraz przeanalizować pojawiający się błąd. Następnie, proszę uzupełnić plik w taki sposób, aby po ponownym uruchomieniu go skryptem run.sh wyświetlał się napis ‘Hello, world’. Proszę nie zmieniać istniejącego kodu, tylko dopisać brakującą część.

Zadanie 2

Celem zadania drugiego jest wstępne zapoznanie studentów z przerwaniami BIOSu na przykładzie wypisywania sekwencji liter. Do wykonania zadania zalecam skorzystanie z przerwania INT 10h, o którym więcej można przeczytać w linku umieszczonym w sekcji trzeciej konspektu.

Treść zadania:

Proszę napisać program wypisujący małe litery od ‘a’ do ‘z’ (obok siebie, bez spacji ani nowych linii), wykorzystując przerwania BIOSu. Wskazówka: do wypisywania znaków można użyć przerwania INT 10h, które zostało częściowo omówione na moim seminarium.

Zadanie 3

Celem zadania trzeciego jest zapoznanie studentów z dodatkowymi przerwaniami BIOSu, umożliwiającymi kontrolę kursora na ekranie, pobieranie danych od użytkownika czy wypisywanie stringa na ekran. Do wykonania zadania polecam użyć przerwań INT 10h oraz INT 16h, dodatkowe informacje na ich temat dostępne są w linkach w sekcji trzeciej konspektu.

Treść zadania:

Proszę napisać program pobierający od użytkownika liczbę. Proces pobrania liczby powinien być poprzedzony wyświetleniem odpowiedniego komunikatu, jak np. “Wprowadz liczbe od 1 do 9: “ lub podobnym. Jeśli użytkownik wprowadzi liczbę od 1 do 9 (przedział obustronnie domknięty), program ma narysować trójkąt o podanej wysokości. Program nie powinien reagować na inne wartości wprowadzane przez użytkownika, wzór rysuje tylko dla liczb z podanego zakresu. Proszę wykorzystać przerwania BIOSu.

Przykładowo, użytkownik wprowadza 4, program rysuje:

```
*
**
***
****
```

3. Linki i materiały dodatkowe

- Bootloader - seminarium video
- Informacje o przerwaniu INT 10h
- Informacje o przerwaniu INT 16h

4. Bibliografia

- https://en.wikipedia.org/wiki/BIOS_interrupt_call#Interrupt_table
- https://stackoverflow.com/
- https://nasm.us/doc/nasmdoc1.html
- https://forum.nasm.us
- https://wiki.osdev.org/Bootloader
- https://en.wikibooks.org/wiki/X86_Assembly