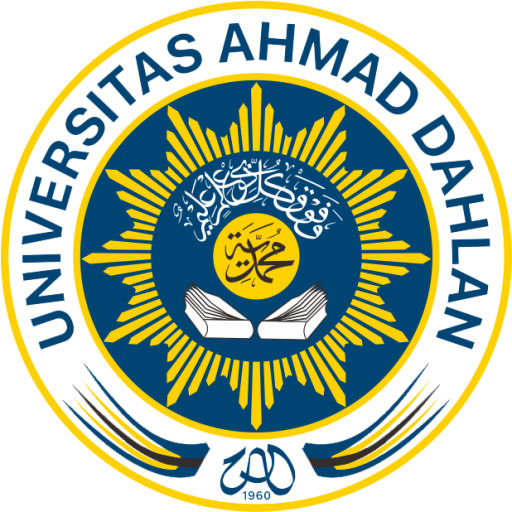
**LAPORAN AKHIR PROJEK**

**MATA KULIAH DASAR SISTEM KOMPUTER**

**APLIKASI PIXEL MOVER**



**Disusun Oleh:**

**Nama : Naurah Mufidah Santoso**

**NIM : 2400018111**

**Kelas : C**

**Link github:** [**https://github.com/NaurahMufidahSantoso/Program-Aplikasi-Pixer-Mover**](https://github.com/NaurahMufidahSantoso/Program-Aplikasi-Pixer-Mover)

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INDUSTRI**

**UNIVERSITAS AHMAD DAHLAN**

**2025**

**1. Ruang Lingkup Aplikasi**

**Nama Aplikasi: Pixel Mover**

* **Tujuan Aplikasi**

**Pixel Mover** bertujuan untuk menampilkan pergerakan kursor mouse dalam mode grafis 320x200 dengan 256 warna pada EMU8086, menggunakan interrupt BIOS untuk mode grafis (0x10) dan pengendalian mouse (0x33).

* **Fitur Utama**
  + Mode Grafis 320x200 dengan 256 warna.
  + Pengendalian Mouse: Menampilkan dan menggerakkan kursor dengan perubahan warna berdasarkan posisi X mouse.
  + Tombol Escape: Program berhenti saat tombol Esc ditekan.
* **Platform yang Didukung**

Dijalankan di EMU8086, emulator untuk prosesor Intel 8086, dengan lingkungan DOS**.**

* **Target Pengguna**
  + Pengguna yang tertarik pada pemrograman grafis dasar di DOS menggunakan Assembly 8086.
  + Pemrogram yang ingin mempelajari interaksi perangkat keras dengan mouse dan grafis di tingkat rendah**.**
* **Batasan (Out of Scope)**
  + Tidak mendukung mode grafis selain 320x200 dengan 256 warna.
  + Tidak ada fitur animasi kursor atau pengolahan suara.
* **Teknologi yang Digunakan**
  + Assembly 8086 untuk pemrograman.
  + Interrupt BIOS 0x10 untuk grafis dan 0x33 untuk pengendalian mouse.

**2. Desain/Rancangan Aplikasi**

|  |
| --- |
| **Gambar 1.1**  Tampilan awal ketika  program di run. |
| **Gambar 1.2**  Cara program berjalan, (Titik bergerak berdasarkan pergerakan mouse dengan warna yang berubah) |
| **Gambar 1.3**  Tampilan grafis yang menunjukkan pergerakan kursor mouse dengan perubahan warna berdasarkan posisi X mouse |

Sketsa Desain (Secara visual, ini akan terlihat seperti layar hitam dengan titik yang bergerak sesuai posisi mouse, dan perubahan warna kursor yang dinamis)

**3. Hasil Akhir Aplikasi**

**A. KODE PEOGRAM**

.MODEL SMALL

.CODE

ORG 100h

proses:

jmp start

oldX dw -1

oldY dw 0

start:

;ubah ke mode grafik

mov ah,00

mov al,13h

int 10h

; reset mouse status saat ini:

mov ax, 0

int 33h

cmp ax, 0

; tampilkan kursor mouse

mov ax, 1

int 33h

check\_mouse\_button:

mov ax,3

int 33h

shr cx,1

cmp bx,1

je not\_required

push cx

push dx

mov cx,oldX

mov dx,oldY

mov ah,0dh

int 10h

; Set warna kursor yang berganti-ganti

; Ganti warna kursor berdasarkan posisi mouse atau urutan

mov al, cl ; Gunakan nilai CX (posisi X mouse) untuk mengganti warna

and al, 0Fh ; Batasi untuk warna 0-15 (PALLETE)

mov ah, 0Ch ; Fungsi untuk menggambar pixel dengan warna

mov bh, 0 ; Page video

int 10h

pop dx

pop cx

not\_required:

mov ah,0dh

int 10h

mov oldX,cx

mov oldY,dx

draw\_pixel:

mov ah,0ch

int 10h

check\_esc\_key:

mov dl,255

mov ah,6

int 21h

cmp al,27

jne check\_mouse\_button

stop:

mov ax,2

int 33h

mov ax,3

int 10h

mov ah,1

mov ch,0

mov cl,8

int 10h

mov dx,offset msg

mov ah,9

int 21h

mov ah,0

int 16h

ret

msg db "press any key. . . $"

end proses

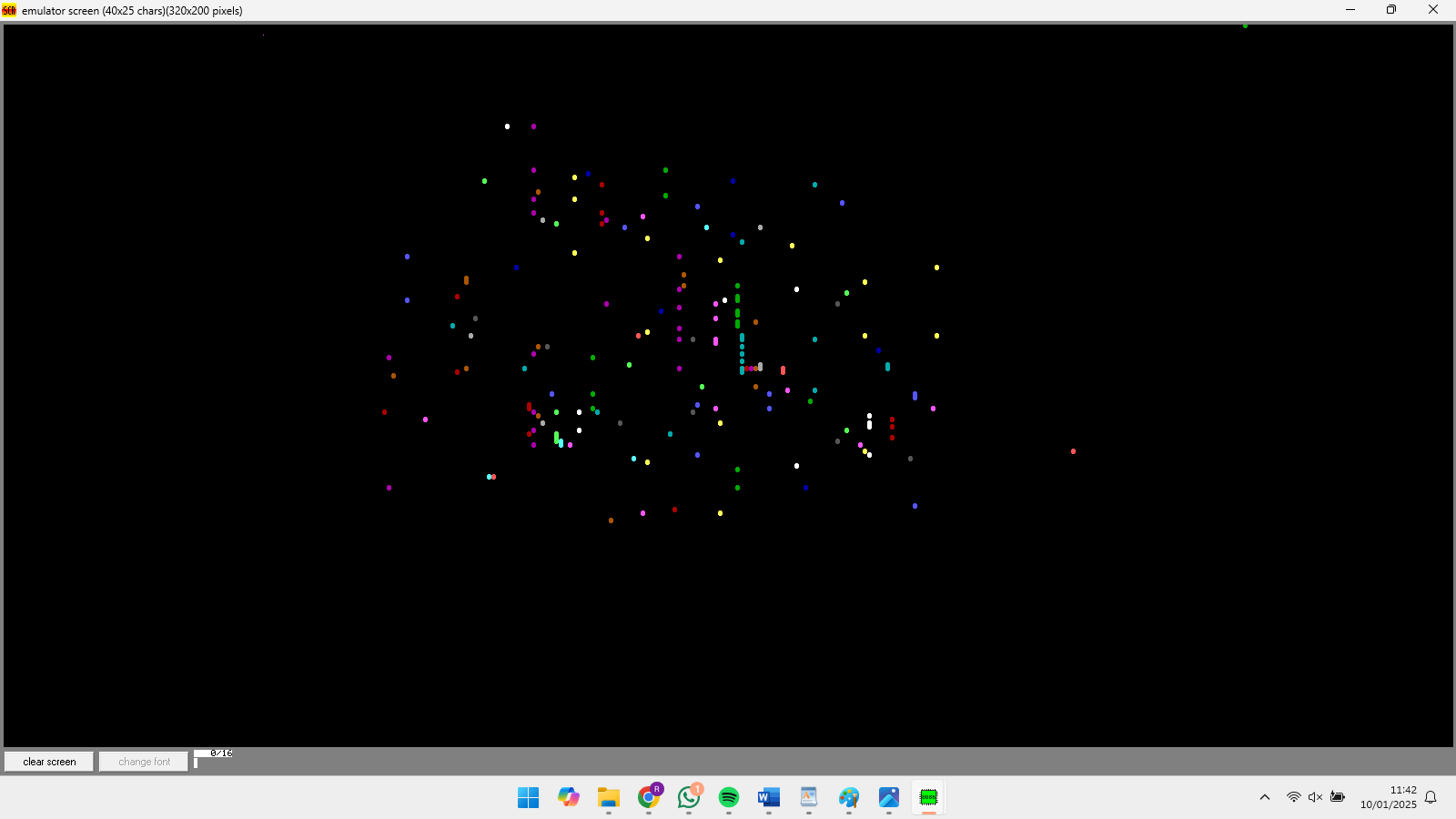
**Penjelasan Kode:**

* **.MODEL SMALL**: Menentukan model memori kecil, cocok untuk program kecil yang tidak menggunakan banyak data.
* .**CODE**: Menandakan awal dari bagian kode program.
* **ORG 100h**: Menentukan alamat awal program, dimulai pada alamat 100h (yang umum untuk program DOS).
* **proses**: adalah label untuk titik awal program.
* **jmp start**: Lompat ke label start untuk memulai eksekusi program.
* **oldX dw -1, oldY dw 0**: Menyimpan posisi kursor sebelumnya (X dan Y). Variabel ini digunakan untuk melacak posisi kursor mouse.
* **mov ah, 00**: Menentukan fungsi BIOS untuk mengubah mode tampilan.
* **mov al, 13h**: Memilih mode grafis 320x200 dengan 256 warna (mode 0x13).
* **int 10h:** Panggil BIOS interrupt 0x10 untuk mengatur mode grafis.
* **mov ax, 0**: Menginisialisasi mouse menggunakan fungsi 0 dari interrupt BIOS 0x33.
* **int 33h**: Panggil BIOS interrupt 0x33 untuk menginisialisasi mouse.
* **mov ax, 1**: Fungsi untuk menampilkan kursor mouse.
* **int 33h**: Memanggil interrupt 0x33 untuk menampilkan kursor mouse.
* **mov ax, 3**: Fungsi untuk membaca posisi dan status tombol mouse.
* **int 33h**: Memanggil interrupt 0x33 untuk mendapatkan posisi kursor dan status tombol mouse.
* **shr cx, 1**: Menggeser bit ke kanan pada register CX untuk memperoleh status tombol mouse (apakah tombol kiri ditekan).
* **cmp bx, 1:** Memeriksa apakah tombol kiri mouse ditekan (status 1 pada BX).
* j**e not\_required:** Jika tombol kiri tidak ditekan, lompat ke not\_required.
* **push cx dan push dx**: Menyimpan nilai CX dan DX (posisi mouse sebelumnya).
* **mov cx, oldX dan mov dx, oldY**: Mengambil posisi mouse sebelumnya dari oldX dan oldY.
* **mov ah, 0dh**: Fungsi BIOS untuk menggambar karakter pada layar.
* **int 10h**: Memanggil interrupt 0x10 untuk menggambar pixel pada posisi lama.
* **mov al, cl**: Menetapkan warna berdasarkan nilai posisi X mouse.
* **and al, 0Fh**: Membatasi warna agar hanya dalam rentang 0-15 (sesuai dengan palet warna 256).
* **mov ah, 0Ch**: Fungsi BIOS untuk menggambar pixel dengan warna tertentu.
* **mov bh, 0**: Halaman video yang digunakan.
* **int 10h**: Menggambarkan pixel pada layar dengan warna yang ditentukan.
* **pop dx pop cx**: Mengembalikan nilai register dx dan cx yang disimpan sebelumnya.
* **mov ah, 0dh**: Menentukan fungsi untuk menggambar pixel.
* **int 10h**: Menggambar pixel pada posisi baru.
* **mov oldX, cx**: Menyimpan posisi X terbaru ke oldX.
* **mov oldY, dx**: Menyimpan posisi Y terbaru ke oldY.
* **mov ah, 0ch**: Fungsi BIOS untuk menggambar pixel.
* **int 10h**: Menggambarkan pixel di layar.
* **mov dl, 255**: Menetapkan dl untuk mengambil input dari keyboard.
* **mov ah, 6:** Fungsi untuk membaca tombol keyboard.
* **int 21h**: Panggil interrupt 0x21 untuk input keyboard.
* **cmp al, 27**: Memeriksa apakah tombol Esc (kode ASCII 27) ditekan.
* **jne check\_mouse\_button**: Jika Esc tidak ditekan, kembali ke pengecekan mouse.
* **mov ax, 2 dan int 33h**: Menonaktifkan mouse.
* **mov ax, 3 dan int 10h**: Mengembalikan mode grafis ke teks (mode 3).
* **mov ah, 1**: Menampilkan kursor dengan fungsi BIOS.
* **mov ch, 0, mov cl, 8, int 10h:** Menampilkan kursor di posisi tertentu.
* **mov dx, offset msg**: Menetapkan alamat pesan yang ingin ditampilkan.
* **mov ah, 9:** Menampilkan pesan.
* **int 21h:** Panggil interrupt 0x21 untuk menampilkan pesan.
* **mov ah, 0**: Menunggu input keyboard.
* **int 16h:** Membaca input dari keyboard.
* **ret:** Mengakhiri program.
* **msg db "press any key. . . $"** : Pesan ini ditampilkan di layar untuk memberi tahu pengguna untuk menekan tombol apa saja untuk melanjutkan atau keluar.

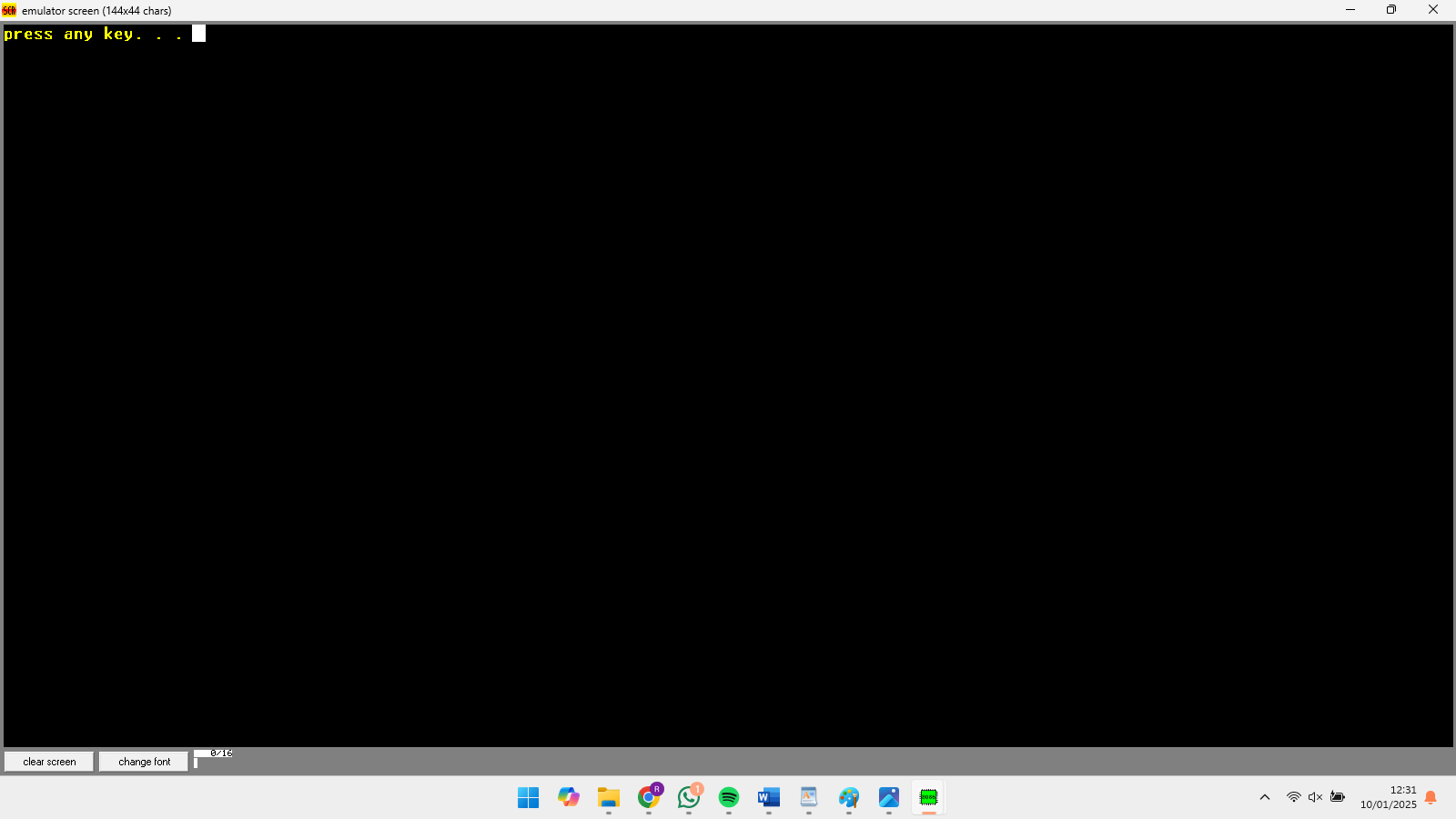
**Diatas adalah kode lengkap dari aplikasi Pixel Mover. Kode ini berfungsi untuk menggambar titik sesuai pergerakan mouse dan mengubah warna kursor berdasarkan posisi X mouse.**

**B. Capture UI dan Penjelasan**

**HASIL OUTPUTNYA**

****

* Titik bergerak: Titik akan mengikuti pergerakan mouse di layar.
* Warna berubah: Warna titik berubah sesuai dengan posisi horizontal (X) kursor, memberikan efek dinamis



* Jika tombol ESC ditekan, aplikasi akan berhenti dan kembali ke mode teks dengan menampilkan pesan "press any key...".

**C.Kesimpulan**

Aplikasi Pixel Mover berhasil memberikan gambaran tentang bagaimana cara berinteraksi dengan tampilan grafis dan mouse menggunakan Assembly dalam emulator EMU8086. Program ini menggambar titik yang bergerak sesuai pergerakan kursor, dengan perubahan warna berdasarkan posisi X mouse. Program ini juga memungkinkan pengguna untuk keluar menggunakan tombol ESC, yang mengembalikan sistem ke mode teks.