|  |  |  |  |
| --- | --- | --- | --- |
| **Lab No** | 10 | **Reg. No** | 224921 |
| **Student Name** | Muhammad Saad Tariq | **Section** | A |

|  |
| --- |
| Readme File |
| The following program performs memory mapping functions of a given physical address in decimal to non- contiguous memory sections of 4KB sized pages. Working of the algorithm:   * Prompts user to enter a physical address in decimal form ranging to a 32-bit integer only. (0 to {232-1}). * The program converts the decimal address into a 32-bit binary address and displays it. * Since the code uses pages of size 4KB (212), we assign the 12 least significant bits as the offset address and the 20 most significant bits as the page number. * The code makes use of simple binary to decimal conversion (of the first 20 bits) to evaluate the page number and display it in decimal form. * The starting address is then calculated in hexadecimal by multiplying the page number by 212 and then converting the decimal to hex form (using the base 16). [Starting address displayed in format 0x\*\*\*\*]. * Lastly the offset is calculated and displayed in decimal by simply converting the remaining 12 bits of the 32-bit physical address from binary to decimal form. |

|  |
| --- |
| Task |
| Code:  #include <stdio.h>  #include <math.h>  int main(){  signed long num, addr, page = 0;  printf("Enter physical address (in decimal): ");  scanf("%ld", &num);  printf("\nPhysical address in decimal: %ld", num);  int bin[32], phy[32], i, count = 0, offset = 0, hex\_count = 0;  char hex[8];  for(i = 0; i < 32; i++)  bin[i] = 0;  for(i = 0; i < 8; i++)  hex[i] = 0;  for(i = 0; num > 0; i++){  bin[i] = num % 2;  num = num / 2;  count++;  }  printf("\n32-bit physical address in binary:\n");  for(i = 0; i < (sizeof(bin)/sizeof(int)); i++){  phy[i] = bin[((sizeof(bin)/sizeof(int)) - 1)-i];  printf("%d ", phy[i]);  }  printf("\n\n");  if(count > 12){    for(i = 0; i < (count - 12); i++){  page += (pow(2, ((count - 12) - i - 1)) \* bin[(count - 1) - i]);  }  printf("Page no. = %ld\n", page);  addr = page \* 4096;  for(i = 0; addr > 0; i++){  signed long temp = 0;  temp = addr % 16;  if(temp < 10)  hex[i] = temp + 48;  else  hex[i] = temp + 55;  addr = addr / 16;  hex\_count++;  }  printf("Starting address = 0x");  for(i = (hex\_count - 1); i >= 0; i--){  printf("%c", hex[i]);  }  printf("\n");  for(i = 0; i < 12; i++){  offset += (pow(2, (11 - i)) \* bin[11 - i]);  }  printf("Offset = %d\n", offset);  }  else{  printf("Page no. = 0\n");  printf("Starting address = 0x0000\n");  for(i = 0; i < count; i++){  offset += (pow(2, (count - 1 - i)) \* bin[count - 1 - i]);  }  printf("Offset = %d\n", offset);  }    return 0;  }  Outputs: |