**UJIAN TENGAH SEMESTER**

**PEMROSESAN DATA TERDISTRIBUSI**

**IF655-D**



**Dikerjakan oleh :**

**Andre Budiman**

**00000032851**

**UNIVERSITAS MULTIMEDIA NUSANTARA**

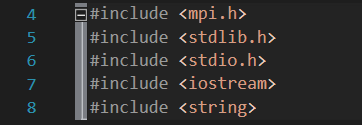
**TANGERANG**

**2020**

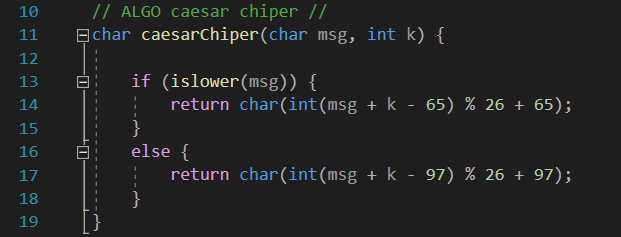
1. Buatlah sebuah program sederhana (tidak lebih dari 500 *lines of code*) yang menyelesaikan sebuah permasalahan *computer science* dengan memanfaatkan pemrosesan / komputasi paralel **[50]**. (CPMK 3-7)

**POTONGAN SOURCE CODE & EKSEKUSI PROGRAM MENGGUNAKAN ALGORITMA CAESAR CIPHER UNTUK PARALLEL PROGRAMMING**

1. Library yang dipakai

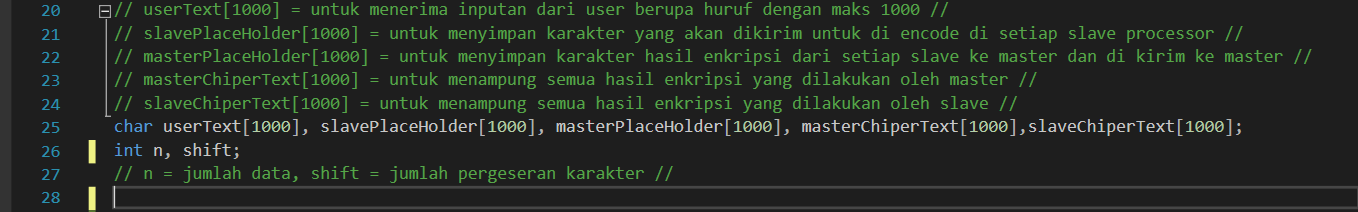


1. Algoritma Caesar Cipher

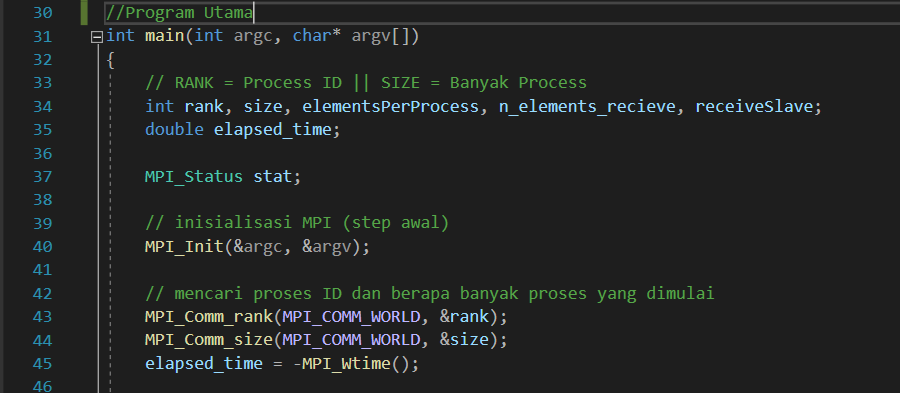


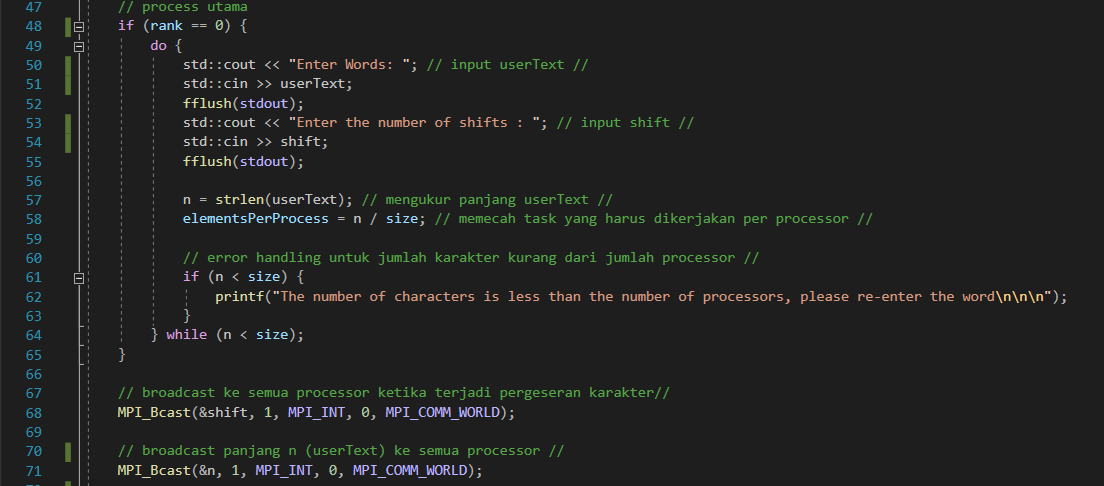
Algoritma ini akan menerima 2 paramter, yaitu: “msg” dan “k”. “msg” akan berfungsi untuk menerima character yang akan di enkripsi oleh sistem, dan “k” adalah jumlah pergeseran yang akan dilakukan pada karakter yang telah diinput.

1. **Inisialisasi Variabel yang akan dipanggil**

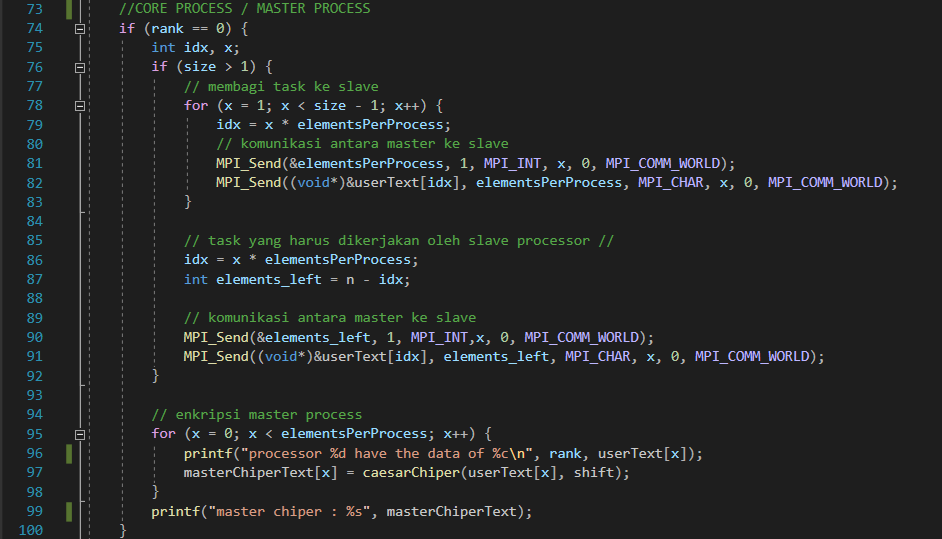


1. **Program Utama**

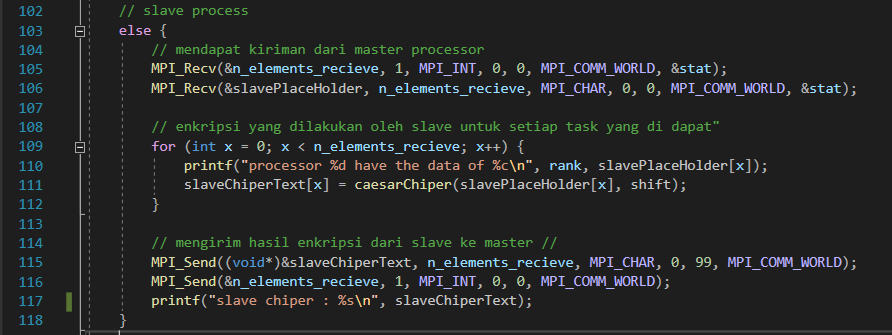




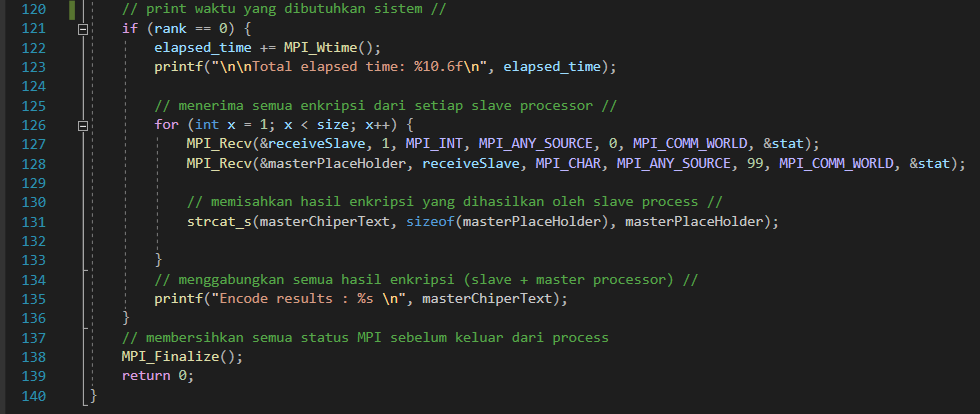
1. **Core Process**



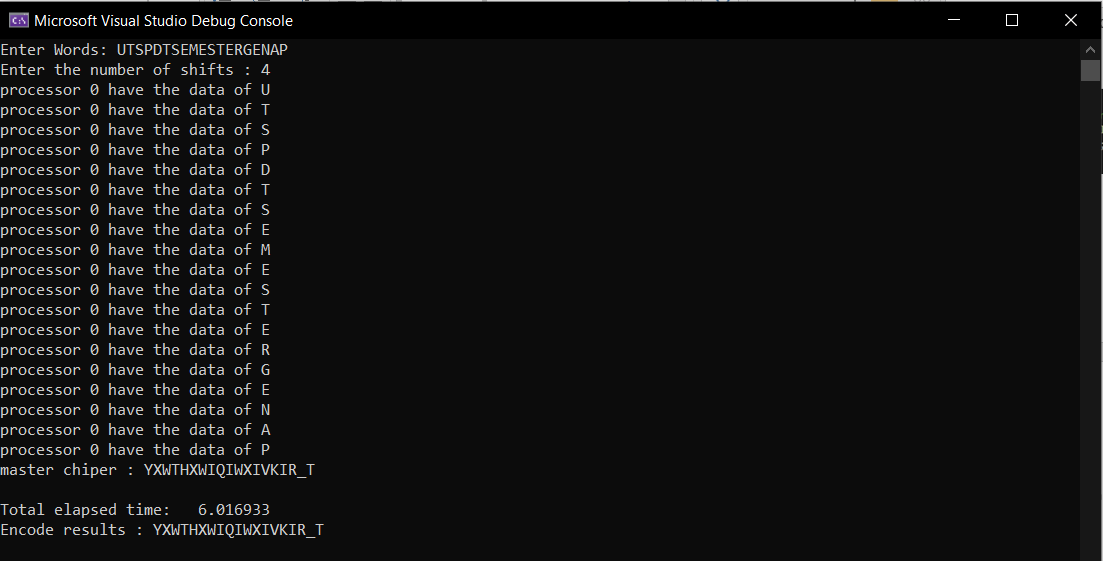
1. **Slave Process**



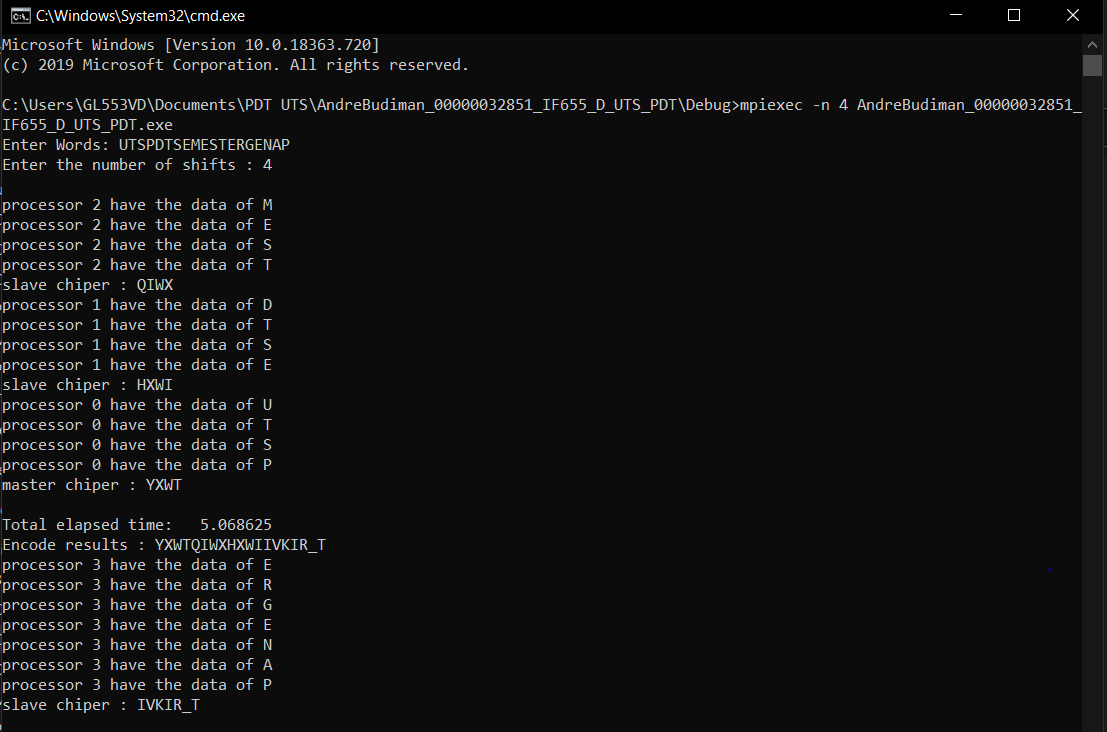
1. **Output Elapsed Time sampai dengan End Code**



1. **Eksekusi Program ( 1 Processor )**



1. **Eksekusi Program ( 4 Processor )**



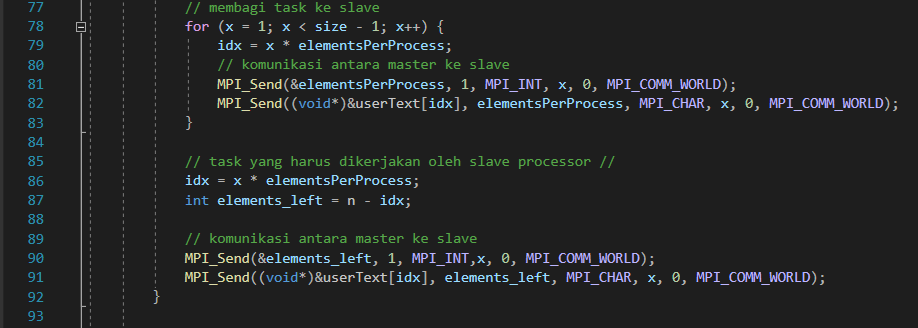
Kesimpulan yang dapat diambil adalah semakin banyak processor yang bekerja, waktu yang dibutuhkan oleh sistem untuk mengerjakan suatu task akan semakin cepat, dapat dilihat perbedaan elapsed time yang dimiliki oleh sistem dengan 1 processor dan 4 processor untuk memproses kata **“UTSPDTSEMESTERGENAP”,** karena dengan adanya 4 processor, task akan dipecah dan dikirimkan ke slave process.

1. Buatlah rancangan metodologi desain Foster untuk program tersebut (*partitioning, communication, agglomeration, mapping*) **[50]**. (CPMK 1-2)
2. **Partitioning**

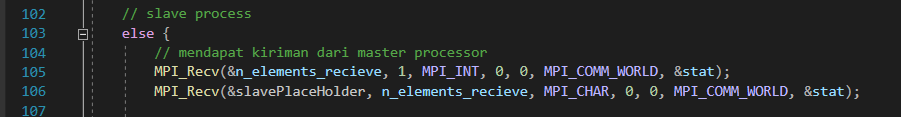
Partitioning merupakan **Domain Decomposition**. Proses utama (master) akan membaca inputan userText dan jumlah task yang diinput oleh user, jumlah task ini berpengaruh terhadap berapa banyak task yang harus dikerjakan oleh sistem (sebanyak n), n sendiri adalah panjang kata yang akan di enkripsi oleh sistem dengan menggunakan notasi “Big O”, jadi setiap karakter yang diinput oleh user merupakan sebuah primitive task, contoh: “UTSPDT” memiliki 6 primitive task.

1. **Communication**

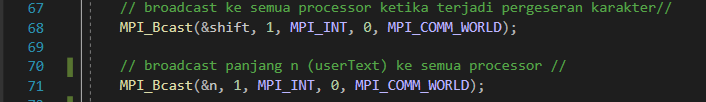
Dalam Communication, dilakukan 2 buah komunikasi, yaitu: Local Communication dan Global Communication. Dalam communication, master process akan mengirimkan character yang telah diinput oleh user untuk di kerjakan oleh setiap process dan jumlah characternya, ketika proses pengiriman ini, kita dapat menggunakan “MPI\_Send” untuk mengirimkan characternya, agar slave dapat menerima character yang dikirimkan oleh master process, kita dapat menggunakan “MPI\_Recv” agar slave process dapat menerimanya. Selain “MPI\_Send” dan “MPI\_Recv”, kita juga dapat menggunakan “MPI\_Bcast” untuk melakukan broadcast agar processor yang berjalan mengetahui jumlah pergeseran character dan panjang katanya.



Komunikasi serta membagi task dari master ke slave



Slave process menerima kiriman task dari master processor

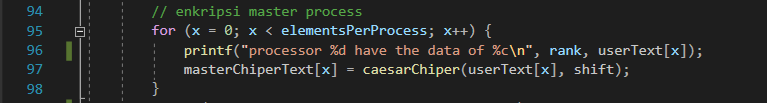


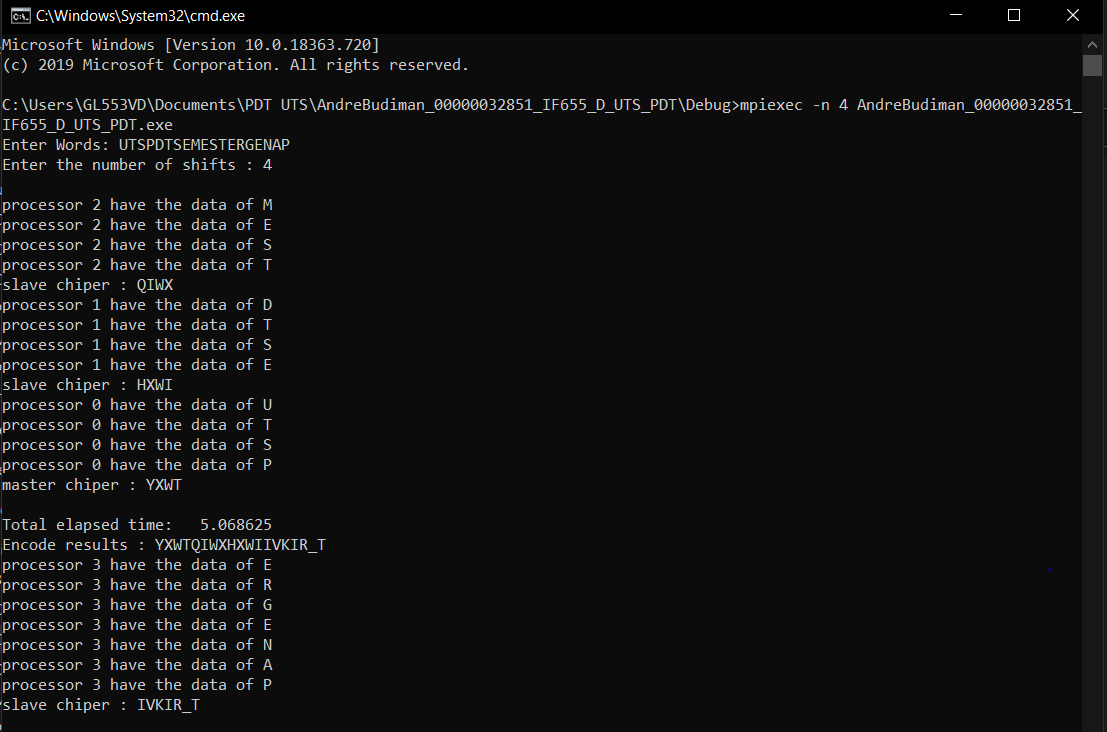
Broadcast pergeseran karakter dan panjang dari userText

1. **Agglomeration**

Agglomeration adalah proses yang menggabungkan antar primitive task agar menjadi 1 task yang akan dikerjakan oleh processor yang diminta oleh user.

Contoh: karakter yang diterima adalah “UTSPDTSEMESTERGENAP”, pada tahap proses ini, processor akan membagi karakter, digunakan 4 processor, sehingga akan terjadi 4 pembagian task. Processor 0 (Master) untuk karakter “UTSP”, processor 1-3 (Slave) untuk karakter “DTSE”,”MEST”, dan “ERGENAP”.



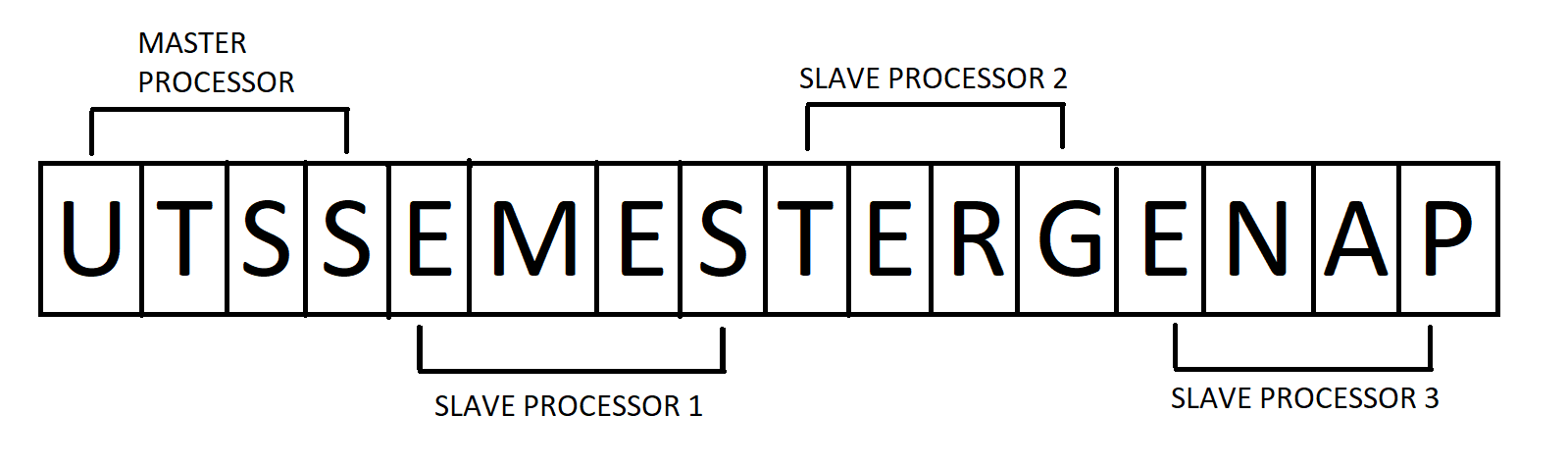


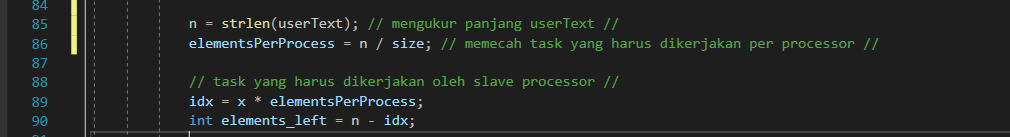
1. **Mapping**

Mapping dilakukan secara dinamis. Proses mapping akan membagi task dan sisa task yang harus dikerjakan. Hal ini dimulai ketika user telah menginput karakter-karakter yang harus dikerjakan oleh processor, lalu processor akan mengecek berapa banyak processor yang dapat bekerja, ketika jumlah processor yang dapat bekerja melebihi 1, maka master processor akan mengirimkannya ke slave processor dengan maksud mempercepat kerja process, hal ini dilakukan juga untuk memotong waktu yang dibutuhkan oleh processor untuk menyelesaikan setiap primitive task. Setiap process yang bekerja juga tidak saling menunggu, atau dengan kata lain, setiap processor dapat bekerja secara bersamaan, hal inilah yang membuat sebuah process yang dipecah dapat menjadi lebih cepat untuk selesai ketika di process lebih dari 1 processor.

Contoh:

Menggunakan 4 processor untuk karakter “UTSSEMESTERGENAP”, terjadi pemecahan primitive task untuk dikirimkan dari master processor ke slave processor, pemecahannya menjadi “UTSS”, “EMES”, “TERG”, dan “ENAP”





Line code 85-86 = pembagian task

Line code 88-90 = sisa task yang harus dikerjakan

Referensi :

1. Quinn, Michael. “Parallel Programming in C with MPI and Open MP Edition 3”.

Source Code:

<https://github.com/andrebudiman/PDT>