หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

บทที่ ๒

อินเตอร์เพรตเตอร์ภาษาเบสิก

ภาษาเบสิก เป็นภาษาคอมพิวเตอร์ที่มีตัวแปลภาษาประเภทอินเตอร์เพรตเตอร์ ตัวแปลภาษาประเภทนี้จะแปลคำสั่งที่พิมพ์เข้ามา และทำงานตามคำสั่งทีละคำสั่ง หลักการทำงานของอินเตอร์เพรตเตอร์คือ การกำหนดรหัสคำสั่งแทนคำสั่งแต่ละคำสั่งและเตรียมการทำงานของแต่ละคำสั่งไว้ในตัวอินเตอร์เพรตเตอร์เอง ทำให้ไม่จำเป็นต้องแปลคำสั่งทั้งหมดเป็นภาษาเครื่องจึงจะทำงานได้ เหมือนตัวแปลภาษาประเภทคอมไพเลอร์

๒.๑ อินเตอร์เพรตเตอร์และคอมไพเลอร์

ภาษาที่ใช้ติดต่อกับคอมพิวเตอร์มีอยู่หลายภาษา แบ่งออกได้เป็น ๒ ประเภท คือ ภาษาระดับต่ำ (Low Level Language) ได้แก่ภาษาแอสเซมบลี และภาษาระดับสูง (High Level Language) ซึ่งแตกแขนงออกเป็นภาษาคอมพิวเตอร์อีกหลาย ๆ ภาษา แต่ละภาษาก็มีความเหมาะสมกับการใช้งานต่าง ๆ ประเภทออกไป ภาษาเหล่านี้ เช่น เบสิก ฟอร์แทรน โคบอล ฯลฯ แต่ไม่ว่าจะเป็นภาษาระดับต่ำหรือภาษาระดับสูง ก็จำเป็นต้องมีตัวแปลภาษาของภาษานั้น ๆ ออกเป็นภาษาเครื่องเสียก่อน จึงจะนำไปสั่งงานคอมพิวเตอร์ให้ทำงานตามที่ต้องการได้

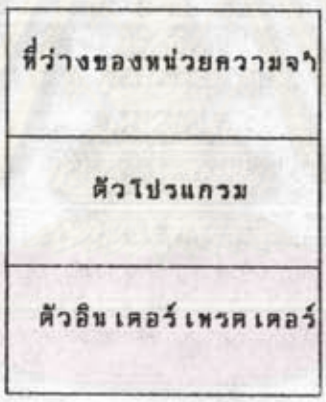
สำหรับภาษาแอสเซมบลีนั้น มีตัวแปลภาษาให้เป็นภาษาเครื่องเรียกว่า แอสเซมเบลอร์ (Assembler) ส่วนภาษาระดับสูงทั้งหลายนั้นมีตัวแปลภาษาให้เป็นภาษาเครื่องอยู่ ๒ ชนิด เรียกว่า อินเตอร์เพรตเตอร์ (Interpreter) และคอมไพเลอร์ (Compiler)

๒.๑.๑ อินเตอร์เพรตเตอร์ การแปลภาษาโดยใช้อินเตอร์เพรตเตอร์นั้น อินเตอร์เพรตเตอร์จะอ่านคำสั่งที่ผู้ใช้พิมพ์เข้ามาทีละบรรทัด และทำงานไปตามคำสั่งทีละ

บรรทัดนั้น แม้ว่าตัวอินเตอร์เพรตเตอร์เองจะถูกเขียนเป็นภาษาเครื่อง แต่ที่จริงอินเตอร์เพรตเตอร์เองจะไม่ทำการแปลภาษาระดับสูงเป็นภาษาเครื่องโดยตรง แต่มันจะทำงานโดยการอ่านโปรแกรมเข้ามาทีละบรรทัด แล้วพยายามทำงานตามคำสั่ง โดยตรวจดูคำสั่งหลักภายในโปรแกรมบรรทัดนั้น มาเปรียบเทียบกับคำสั่งที่เครื่องถูกสอนเอาไว้แล้วว่าหมายถึงให้ไปทำอะไร เมื่อใช้งานอินเตอร์เพรตเตอร์ ตัวโปรแกรมและตัวอินเตอร์เพรตเตอร์จะต้องอยู่ในหน่วยความจำพร้อมกันเสมอ ดังรูปที่ ๒.๑

ตัวอย่างภาษาที่ใช้ตัวแปลภาษาแบบอินเตอร์เพรตเตอร์ คือ ภาษาเบสิก

หน่วยความจำ



รูปที่ ๒.๑ แสดงการอยู่ในหน่วยความจำของโปรแกรมที่เขียนภายใต้อินเตอร์เพรตเตอร์

๒.๑.๒ คอมไพเลอร์ การแปลภาษาโดยใช้คอมไพเลอร์นั้นคอมไพเลอร์จะแปลคำสั่งทั้งหมดในโปรแกรมออกเป็นภาษาเครื่อง ซึ่งพร้อมจะใช้สั่งงานคอมพิวเตอร์ โดยไม่ต้องอาศัยคอมไพเลอร์อีกต่อไป ดังรูปที่ ๒.๒

หน่วยความจำ

ที่ว่างของหน่วยความจำ
ตัวโปรแกรมที่ถูกแปลเป็นภาษาเครื่อง

รูปที่ ๒.๒ แสดงการอยู่ในหน่วยความจำของโปรแกรมที่เขียนภายใต้คอมไพเลอร์

๒.๒ ลักษณะการทำงานของอินเตอร์เพรตเตอร์ภาษาเบสิก

ตัวแปลภาษาเบสิก หรือตัวแปลภาษาชนิดอินเตอร์เพรตเตอร์ ซึ่งมีลักษณะการทำงานอยู่ ๒ แบบ ดังนี้คือ

๒.๒.๑ แบบทำงานทันที (Direct Mode) อินเตอร์เพรตเตอร์จะแปลคำสั่งทีละคำสั่งแล้วทำงานตามคำสั่งนั้นทันที เช่น ถ้าผู้ใช้ป้อนคำสั่ง PRINT X เข้าไป อินเตอร์-เพรตเตอร์จะไปหาค่าของตัวแปร X และพิมพ์ค่าของตัวแปร X ออกมาทันที คำสั่งที่สั่งให้อินเตอร์เพรตเตอร์ทำงานแบบทันทีจะไม่ถูกเก็บไว้ในหน่วยความจำ ดังนั้นทุกครั้งที่ผู้ใช้ต้องการสั่งงานแบบเดิมอีก ผู้ใช้จำเป็นต้องป้อนคำสั่งเดิมเข้าไปใหม่ทุกครั้ง

๒.๒.๒ แบบทำงานไม่ทันที (Indirect Mode) อินเตอร์เพรตเตอร์จะเก็บชุดของคำสั่งที่เรียกว่า โปรแกรม (Program) ไว้ในหน่วยความจำก่อน โดยแต่ละบรรทัดของคำสั่งต้องมีเลขหมายบรรทัด (Line Number) อยู่ประจำบรรทัดของคำสั่ง เมื่อต้องการให้โปรแกรมเริ่มต้นทำงาน ต้องป้อนคำสั่ง "RUN" อินเตอร์เพรตเตอร์จะเริ่มต้นแปลคำสั่งทีละบรรทัดตั้งแต่บรรทัดที่มีเลขหมายบรรทัดต่ำสุด และทำงานตามคำสั่งบรรทัดนั้นตามลำดับจนกระทั่งจบโปรแกรม

คำสั่งบางคำสั่งเป็นคำสั่งแบบทำงานทันทีเท่านั้น เช่น LIST, NEW, AUTO เป็นต้น และคำสั่งบางคำสั่งก็เป็นคำสั่งแบบทำงานไม่ทันทีเท่านั้น เช่น DATA, READ เป็นต้น ขณะที่บางคำสั่งเป็นได้ทั้งคำสั่งแบบทำงานทันทีและไม่ทันที เช่น PRINT, HOME, KILL เป็นต้น

๒.๓ <u>รหัสคำสั่ง</u> (TOKEN)

เนื่องจากภาษาเบสิกเป็นภาษาระดับสูง เพื่อให้คำสั่งต่าง ๆ สื่อความหมายเป็นที่เข้าใจและจดจำได้ง่ายจึงต้องประกอบด้วยตัวอักษรหลายตัว เช่น คำสั่ง PRINT หมายถึงให้พิมพ์ค่าออกมา เป็นต้น เมื่อผู้ใช้เขียนโปรแกรมซึ่งจำเป็นต้องเก็บไว้ในหน่วยความจำ ถ้าอินเตอร์เพรตเตอร์เก็บคำสั่งต่าง ๆ ในรูปของตัวอักษร จะทำให้สิ้นเปลืองเนื้อที่หน่วยความจำ และยังทำให้เสียเวลาในการแปลคำสั่งอีกด้วย ดังนั้นอินเตอร์เพรตเตอร์จึงกำหนดรหัสขึ้นมาแทนคำสั่งแต่ละคำสั่ง เรียกว่า รหัสคำสั่ง (TOKEN) รหัสคำสั่งจะมีความยาว ๑ หรือ ๒ ไบต์ ขึ้นอยู่กับประเภทของคำสั่ง คือ คำสั่งที่ไม่เป็นฟังก์ชั่น (Commands หรือ Statements) จะมีความยาว ๑ ไบต์ ตัวอย่างคำสั่งประเภทนี้ เช่น PRINT,AUTO เป็นต้น ส่วนคำสั่งที่เป็นฟังก์ชั่น (Functions) จะมีความยาว ๒ ไบต์ ตัวอย่างคำสั่งประเภทนี้ เช่น ABS, ASC เป็นต้น และเพื่อแบ่งแยกระหว่างคำสั่งกับข้อมูลที่ไม่เป็นคำสั่งซึ่งถูกเก็บในหน่วยความจำ รหัสคำสั่งทุกตัวจะมีบิต (Bit) ทางด้านซ้ายสุดของแต่ละไบต์เป็น ๑ เสมอ เช่น คำสั่ง PRINT จะมีรหัสคำสั่งเท่ากับ 91H คำสั่ง ABS จะมีรหัสคำสั่งเท่ากับ FF86H เป็นต้น

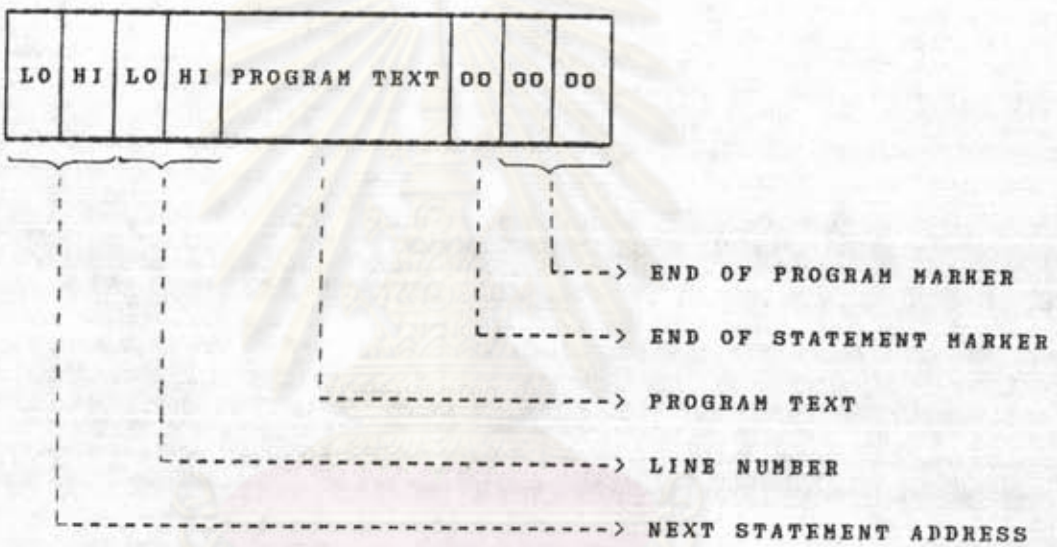
๒.๔ <u>การเก็บโปรแกรมและข้อมูลในหน่วยความจำ</u>

โปรแกรมที่ผู้ใช้เขียนขึ้นนอกจากจะประกอบด้วยคำสั่งต่าง ๆ แล้ว ยังประกอบด้วยส่วนอื่นอีกหลาย ๆ ส่วน ได้แก่ เลขหมายบรรทัด, ชื่อตัวแปรต่าง ๆ, ค่าคงที่ตัวเลขและตัวอักษร เป็นต้น และเมื่อโปรแกรมเริ่มต้นทำงาน ถ้ามีการกำหนดค่าให้ตัวแปร ก็จะมีการเก็บค่าของตัวแปรไว้ในหน่วยความจำเพิ่มขึ้นอีกด้วย เพื่อเป็นการประหยัดเนื้อที่ของ

หน่วยความจำ การเก็บโปรแกรมและข้อมูลจึงต้องมีการจัดรูปแบบให้เหมาะสมดังนี้

๒.๔.๑ การเก็บโปรแกรมในหน่วยความจำ

โปรแกรมที่ผู้ใช้เขียนขึ้น จะถูกจัดเก็บในหน่วยความจำส่วนที่เรียกว่า เนื้อที่โปรแกรม (Program Text Area) แต่ละบรรทัดของคำสั่งจะถูกจัดเรียงตามลำดับของเลขหมายบรรทัดติดต่อกันไป โดยแต่ละบรรทัดของคำสั่งมีรายละเอียดดังรูปที่ ๒.๓



รูปที่ ๒.๓ รูปแบบบรรทัดคำสั่ง

๒.๔.๑.๑ ตำแหน่งคำสั่งบรรทัดถัดไป (Next Statement Address) มีความยาว ๒ ไบต์ เพื่อเก็บตำแหน่งของคำสั่งบรรทัดต่อไป โดยไบต์ที่ ๑ เป็นไบต์ที่มีลำดับตำแหน่งต่ำ (Low Order Byte) ไบต์ที่ ๒ เป็นไบต์ที่มีลำดับตำแหน่งสูง (High Order Byte) มีการจัดเก็บเป็นเลขฐานสิบหก

๒.๔.๑.๒ เลขหมายบรรทัด มีความยาว ๒ ไบต์ เพื่อเก็บเลขหมายบรรทัดของคำสั่ง โดยไบต์ที่ ๑ เป็นไบต์ที่มีลำดับตำแหน่งต่ำ ไบต์ที่ ๒ เป็นไบต์ที่มีลำดับตำแหน่งสูง มีการจัดเก็บเป็นเลขฐานสิบหก

๒.๔.๑.๓ บรรทัดของโปรแกรม (Program Text) เพื่อเก็บรหัสคำสั่ง รวมทั้งตัวอักษรที่ประกอบกับคำสั่งนั้น ซึ่งได้แก่ ชื่อตัวแปร, ค่าคงที่ตัวเลขและตัวอักษร เป็นต้น

๒.๔.๑.๔ เครื่องหมายแสดงการหมดบรรทัด (End of Statemant Marker) ใช้เลขฐานย์จำนวน ๑ ไบต์

๒.๔.๑.๕ เครื่องหมายแสดงการหมดโปรแกรม (End of Program Marker) ใช้เลขฐานย์จำนวน ๒ ไบต์

ตัวอย่างโปรแกรมภาษาเบสิกและการเก็บโปรแกรมในหน่วยความจำ แสดงดังรูปที่ ๒.๔ และ ๒.๕ ตามลำดับ

```
100 A = 1
110 B = 2
120 C = A+B
130 PRINT A,B,C
```

รูปที่ ๒.๔ ตัวอย่างโปรแกรมที่ต้องการเก็บในหน่วยความจำ

	๑	๒			๓			๔							
628F	99	62	64	00	41	20	F0	20	12	00	A3	62	6E	00	42	20
629F	F0	20	13	00	AF	62	78	00	43	20	F0	20	41	F2	42	00
62AF	BB	62	82	00	91	20	41	2C	42	2C	43	00	๕ 00	00		

รูปที่ ๒.๕ แสดงการเก็บโปรแกรมรูปที่ ๒.๔ ในหน่วยความจำ

อธิบายหมายเลขที่แสดงในรูปที่ ๒.๔ ได้ดังนี้

หมายเลข ๑ แสดง ตำแหน่งคำสั่งบรรทัดถัดไปของคำสั่งบรรทัด 100 ซึ่งเก็บค่า "6299" เป็นตำแหน่งของคำสั่งเลขหมายบรรทัด 110

หมายเลข ๒ แสดง เลขหมายบรรทัดของคำสั่งบรรทัด 100 ซึ่งเก็บค่า 0064H มีค่าเท่ากับเลข 100 ฐานสิบ

หมายเลข ๓ แสดง บรรทัดของโปรแกรมของคำสั่งบรรทัด 100 คือ ข้อความต่าง ๆ ภายในเลขหมายบรรทัด 100

หมายเลข ๔ แสดง เครื่องหมายการหมดบรรทัดของคำสั่งบรรทัด 100 คือ "00"

หมายเลข ๕ แสดง เครื่องหมายการหมดโปรแกรม คือ "0000"

๒.๔.๒ การเก็บข้อมูลในหน่วยความจำ

เมื่อโปรแกรมเริ่มต้นทำงาน และมีการกำหนดค่าตัวเลขหรือค่าสตริงให้กับตัวแปร จะมีการเก็บชื่อของตัวแปรและค่าของตัวแปรไว้ในหน่วยความจำที่เรียกว่าตารางตัวแปร (Variable Table) และตารางค่าสตริง (String Table) ตารางตัวแปรเป็นตารางที่ใช้เก็บชื่อตัวแปรและค่าของตัวแปรตัวเลข ส่วนตารางสตริงเป็นตารางที่ใช้เก็บค่าของตัวแปรสตริง

๒.๔.๒.๑ ตารางตัวแปร มีรูปแบบสำหรับตัวแปรหนึ่ง ๆ ดังนี้

๒.๔.๒.๑.๑ ประเภทของตัวแปร ใช้ ๑ ไบต์ มีค่าและความหมายดังนี้

๒ หมายถึงตัวแปรจำนวนเต็ม (Integer Variable)

๓ หมายถึงตัวแปรสตริง (String Variable)

๔ หมายถึงตัวแปรทศนิยม ๗ ตำแหน่ง (Single Precision Variable)

๘ หมายถึงตัวแปรทศนิยม ๑๖ ตำแหน่ง (Double Precision Variable)

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

๒.๔.๒.๑.๒ ชื่อของตัวแปร มี ๒ กรณี คือ

กรณีที่ ๑ ถ้ายาวไม่เกิน ๒ ไบต์ จะใช้ ๓ ไบต์ โดยที่ไบต์ที่ ๑,๒ เป็นรหัสแอสกีของชื่อตัวแปรนั้น และไบต์ที่ ๓ เป็นศูนย์เสมอ

เช่น ตัวแปร A จะเก็บเป็น 410000

กรณีที่ ๒ ถ้ายาวมากกว่า ๒ ไบต์ จะใช้จำนวนไบต์เท่ากับจำนวนตัวอักษรของชื่อตัวแปร + ๑ โดยไบต์ที่ ๑,๒ เป็นรหัสแอสกีของชื่อตัวแปร ๒ ตัวแรก ไบต์ที่ ๓ บอกว่าความยาวยาวกว่า ๒ ไบต์เท่าใด และตั้งแต่ไบต์ที่ ๔ เป็นต้นไปเป็นค่ารหัสแอสกีของชื่อตัวแปรที่เหลือ โดยมีบิตซ้ายสุดของแต่ละไบต์เป็น ๑

เช่น ตัวแปร ABCD จะเก็บเป็น 414202C3C4

๒.๔.๒.๑.๓ ค่าของตัวแปรตัวเลข หรือตำแหน่งของค่าตัวแปรสตริง

กรณีที่ ๑ ตัวแปรตัวเลข ค่านี้จะเป็นค่าของตัวแปร ความยาวที่ใช้เก็บเป็นดังนี้

๒ ไบต์ สำหรับตัวแปรจำนวนเต็ม

๔ ไบต์ สำหรับตัวแปรทศนิยม ๗ ตำแหน่ง

๘ ไบต์ สำหรับตัวแปรทศนิยม ๑๖ ตำแหน่ง

กรณีที่ ๒ ตัวแปรสตริง ค่านี้จะเป็นตำแหน่งของค่าของตัวแปร ความยาวที่ใช้เก็บเท่ากับ ๒ ไบต์

๒.๔.๒.๑.๔ ความยาวของค่าของตัวแปรสตริง สำหรับกรณีที่เป็นตัวแปรสตริงเท่านั้น เพื่อบอกให้ทราบว่าค่าของตัวแปรมีความยาวเท่าใด

๒.๔.๒.๒ ตารางค่าสตริง เป็นตารางที่ใช้เก็บค่าของตัวแปรสตริง โดยจะเก็บค่าของตัวแปรสตริงทุกครั้งที่มีการกำหนดค่าใหม่ให้ตัวแปรสตริงและปรับตำแหน่งกับความยาวของค่าของตัวแปรสตริงในตารางตัวแปรควบคู่ไปด้วย

๒.๕ การแปลภาษาของอินเตอร์เพรตเตอร์และตารางที่จำเป็น

การเขียนโปรแกรมสั่งงานเครื่องคอมพิวเตอร์ด้วยภาษาเบสิก เมื่อจบการเขียน

012938

คำสั่งแต่ละบรรทัด อินเตอร์เพรตเตอร์จะทำการแปลงคำสั่งและนำไปเก็บในหน่วยความจำ หลังจากเขียนจบโปรแกรมและต้องการให้โปรแกรมเริ่มต้นทำงาน ต้องป้อนคำสั่ง "RUN" อินเตอร์เพรตเตอร์จะเริ่มต้นอ่านโปรแกรมเข้ามาทีละบรรทัด แล้วพยายามทำงานตามคำสั่ง โดยตรวจดูคำสั่งหลักภายในโปรแกรมบรรทัดนั้นมาเปรียบเทียบกับคำสั่งที่เครื่องถูกสอนเอาไว้แล้วว่าหมายถึงให้ไปทำอะไร ขั้นตอนการแปลงคำสั่งและการถอดรหัสคำสั่งเพื่อทำงานตามจุดประสงค์ที่เขียนโปรแกรมของอินเตอร์เพรตเตอร์ จำเป็นต้องอาศัยตารางดังต่อไปนี้

๒.๔.๑ ตารางคำสั่ง (Statement Table)

เป็นตารางที่เก็บคำสั่งทั้งหลายของอินเตอร์เพรตเตอร์ ตารางนี้อยู่ที่ตำแหน่ง 0252H-04D7H แต่ละคำสั่งมีความยาวไม่คงที่ขึ้นอยู่กับจำนวนของตัวอักษรในแต่ละคำสั่ง มีรูปแบบดังนี้

[คำสั่งที่ตัดอักษรตัวแรกทิ้ง และบิตซ้ายสุดของอักษรตัวสุดท้ายเป็น ๑ +
รหัสคำสั่ง (๑ ไบต์)]

การที่ตัดอักษรตัวแรกทิ้งได้ เนื่องจากคำสั่งที่มีตัวอักษรตัวแรกเหมือนกันจะอยู่เรียงลำดับกันไป เมื่อหมดทุกคำสั่งของแต่ละกลุ่มอักษรจะมี "00" ต่อท้ายดังรูปที่ ๒.๖

ในการแปลงคำสั่งหนึ่ง ๆ ให้เป็นรหัสคำสั่ง อินเตอร์เพรตเตอร์ใช้วิธีเปรียบเทียบทีละคำสั่ง เริ่มจากคำสั่งแรกจนกระทั่งคำสั่งสุดท้ายของกลุ่มอักษร สำหรับการเปรียบเทียบแต่ละคำสั่ง อินเตอร์เพรตเตอร์ใช้วิธีเปรียบเทียบทีละไบต์ แต่เนื่องจากรูปแบบของตารางคำสั่งไม่ได้เก็บความยาวของคำสั่งไว้ อินเตอร์เพรตเตอร์จึงใช้บิตซ้ายสุดของแต่ละไบต์เป็นตัวกำหนดความยาวแทน โดยการเปรียบเทียบ ถ้าพบว่าเท่ากันทุกไบต์รวมทั้งไบต์ที่บิตซ้ายสุดเป็น ๑ ไบต์ถัดไปจะเป็นรหัสคำสั่ง มิฉะนั้นอินเตอร์เพรตเตอร์จะเปรียบเทียบคำสั่งต่อไปจนกระทั่งพบค่า "00" แสดงว่าเกิดความผิดพลาดของคำสั่ง (Syntax Error)

รูปที่ ๒.๖ แสดงตารางคำสั่ง

รหัสคำสั่งทั้ง ๒ ประเภท มีรูปแบบการเก็บในตารางคำสั่งดังนี้

๒.๔.๑.๑ รหัสคำสั่งที่ไม่เป็นฟังก์ชัน

จะหราบได้โดยสังเกตจากบิตซ้ายสุดของรหัสคำสั่งจะเป็น ๑ ค่ารหัสนี้เป็นรหัสที่แท้จริงของคำสั่งนั้น

เช่น BFH เป็นรหัสของคำสั่ง FILES เป็นต้น

๒.๔.๑.๒ รหัสคำสั่งที่เป็นฟังก์ชัน

จะหราบได้โดยสังเกตจากบิตซ้ายสุดของรหัสคำสั่งจะเป็น ๐ ดังนั้นต้องนำรหัสที่ได้บวกกับค่า FF๘๐H จึงจะเป็นรหัสที่แท้จริงของคำสั่ง

เช่น ๐๙H เป็นรหัสของคำสั่ง SIN มีรหัสที่แท้จริงเป็น FF๘๙H เป็นต้น

๒.๔.๒ ตารางเวกเตอร์ของคำสั่ง (Statement Vector Table)

เป็นตารางที่เก็บตำแหน่งเริ่มต้นของกลุ่มอักษรแต่ละกลุ่มในตารางคำสั่ง ตารางนี้อยู่ที่ตำแหน่ง ๐๒๑EH - ๐๒๕๑H แต่ละเวกเตอร์ของคำสั่งมีความยาวคงที่ ๒ ไบต์ ดังรูปที่ ๒.๗

๐๒๑๐														๕๒	๐๒
๐๒๒๐	๖๓	๐๒	๖E	๐๒	AD	๐๒	D๙	๐๒	FE	๐๒	๑๔	๐๓	๒๘	๐๓	๔C	๐๓
๐๒๓๐	๖C	๐๓	๖D	๐๓	๗๒	๐๓	A๖	๐๓	BF	๐๓	DB	๐๓	EE	๐๓	๐C	๐๔
๐๒๔๐	๐D	๐๔	๔B	๐๔	๘๔	๐๔	๙B	๐๔	A๔	๐๔	BA	๐๔	D๒	๐๔	D๖	๐๔
๐๒๕๐	D๗	๐๔	<----- ตำแหน่งของกลุ่มอักษรตัว "Z"													

รูปที่ ๒.๗ แสดงตารางเวกเตอร์ของคำสั่ง

อินเตอร์เพรตเตอร์ใช้ตารางเวกเตอร์ของคำสั่ง เพื่อหาตำแหน่งเริ่มต้นของกลุ่มอักษรของคำสั่งที่ต้องการแปลงเป็นรหัสคำสั่งในตารางคำสั่ง ทำให้สามารถลดขนาดของตารางคำสั่งและเพิ่มความรวดเร็วในการแปลงคำสั่งให้เป็นรหัสคำสั่ง

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

๒.๕.๓ ตารางตำแหน่งการทำงานของคำสั่ง (Subroutine Vector Table)

เป็นตารางที่เก็บค่าตำแหน่งของการทำงานของแต่ละคำสั่ง ตารางนี้อยู่ที่ตำแหน่ง 0108H-021BH มีความยาวคงที่คือ ๒ ไบต์ สำหรับแต่ละตำแหน่งของคำสั่งในการทำงานของแต่ละคำสั่งอินเตอร์เพรตเตอร์จะอ่านโปรแกรมเข้ามาทีละบรรทัด และนำรหัสคำสั่งไปผ่านการคำนวณ หาให้ได้ตำแหน่งในตารางตำแหน่งการทำงานของคำสั่งซึ่งเก็บตำแหน่งการทำงานของรหัสคำสั่งนั้นอีกทีหนึ่ง หลังจากนั้นจึงเริ่มต้นทำงานตามคำสั่งนั้น

การคำนวณหาตำแหน่งที่เก็บตำแหน่งการทำงานของคำสั่งเป็นดังนี้

ตำแหน่ง = 108H + (รหัสคำสั่งในรูปฐานสิบหก - 81H) * 2H

```
0100 C3 51 5E F4 2B 55 2C 00 D4 45 AA 12 50 47 CF 15 .Q^.+U,..E..PG..
0110 13 19 AE 3B D2 19 F6 15 60 15 34 15 1F 17 B5 45 ...;....`.4....E
0120 48 15 B4 15 D1 15 CF 45 65 17 C6 46 C6 20 F4 44 H......Ee..F. .D
0130 5D 16 A1 1E CC 22 29 46 92 0D 92 0D 5D 17 C1 20 ]....")F....]..
0140 62 20 D1 15 3D 46 3E 46 43 46 81 46 E8 3E E1 16 b ..=F>FCF.F.>..
0150 A7 16 8A 22 EC 16 FD 22 9F 14 A2 14 A5 14 A8 14 ...".."."......
0160 A6 18 96 15 1C 4E 3D 4E AE 4E 06 52 CF 15 37 4F .....N=N.N.R..7O
0170 2E 24 78 24 64 5A 71 59 59 55 E0 5B DF 5B 1C 55 .$x$dZqYYU.[.[.U
0180 FE 53 A7 54 B1 5A 1E 59 87 5A BC 55 BB 55 E0 54 .S.T.Z.Y.Z.U.U.T
0190 6E 5A D9 25 C3 25 5B 25 B2 25 CE 25 D1 25 09 26 nZ.%.%[%.%.%.%.&
01A0 5D 26 93 26 AA 26 E1 26 0F 28 0F 28 0F 28 11 27 ]&.&.&.&.(.(.(.'
01B0 3D 27 D3 4A 03 4B 0C 4B FC 2A F8 2C E7 2A 08 39 ='.J.K.K.*.,.*.9
01C0 18 3A B8 3A 4A 29 66 39 B2 3A 45 3B 5A 3B 71 4C .:.:J)f9.:E;Z;qL
01D0 49 1E 67 4A 37 48 2D 4B 73 4A 83 4A C2 22 BA 4A I.gJ7H-KsJ.J."."J
01E0 2D 48 32 48 44 1E F4 2B 6C 2C 98 2C E5 2C 00 00 -H2HD..+l,.,.,..
01F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0200 00 00 00 00 F8 52 FB 52 FE 52 00 00 C8 56 79 57 .....R.R.R...VyW
0210 90 57 DF 52 E2 52 E5 52 07 27 5E 27 ED 26 52 02 .W.R.R.R.'^'.&R.
0220 63 02 6E 02 AD 02 D9 02 FE 02 14 03 28 03 4C 03 c.n.........(.L.
0230 6C 03 6D 03 72 03 A6 03 BF 03 DB 03 EE 03 0C 04 l.m.r...........
0240 0D 04 48 04 84 04 9B 04 A4 04 BA 04 D2 04 D6 04 ..K.............
0250 D7 04 4E C4 F7 42 D3 06 54 CE 0E 53 C3 14 55 54 ..N..B..T..S..UT
0260 CF A7 00 55 54 54 4F CE 36 45 45 D0 D4 00 4C 4F ...UTTO.6EE...LO
0270 53 C5 8C 4F 4E D4 98 4C 45 41 D2 92 49 4E D4 1B S..ON..LEA..IN..
0280 53 4E C7 1C 44 42 CC 1D 56 C9 2A 56 D3 2B 56 C4 SN..DB..V.*V.+V.
0290 2C 4F D3 0C 48 52 A4 15 41 4C CC B1 4F 4D 4D 4F ,O..HR..AL..OMMO
02A0 CE B3 48 41 49 CE B4 4F 4C 4F D2 CD 00 41 54 C1 ..HAI..OLO...AT.
02B0 84 49 CD 86 45 46 53 54 D2 A9 45 46 49 4E D4 AA .I..EFST..EFIN..
02C0 45 46 53 4E C7 AB 45 46 44 42 CC AC 45 C6 96 45 EFSN..EFDB..E..E
02D0 4C 45 54 C5 A6 45 CC A6 00 4E C4 81 4C 53 C5 9E LET..E...N..LS..
02E0 52 41 53 C5 A2 44 49 D4 A3 52 52 4F D2 A4 52 CC RAS..DI..RRO..R.
02F0 E5 52 D2 E6 58 D0 0B 4F C6 2E 51 D6 FA 00 4F D2 .R..X..O..Q...O.
```

ตำแหน่งการทำงานของคำสั่ง CLOSE

คำสั่ง CLOSE มีรหัสคำสั่งเป็น BCH

รูปที่ ๒.๔ แสดงการหาตำแหน่งการทำงานของคำสั่ง "CLOSE"

เช่น คำสั่ง CLOSE มีรหัสคำสั่งเป็น BCH จะมีตำแหน่งที่เก็บตำแหน่งการทำงานของคำสั่งอยู่ที่ 017EH ซึ่งคำนวณจาก 108H + (BCH - 81H) * 2H ซึ่งที่ตำแหน่ง 017EH ของตารางตำแหน่งการทำงานของคำสั่งเก็บค่า 551CH ดังนั้นตำแหน่งการทำงานของคำสั่ง CLOSE เริ่มที่ตำแหน่ง 551CH เป็นต้น ดังรูปที่ ๒.๘

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย