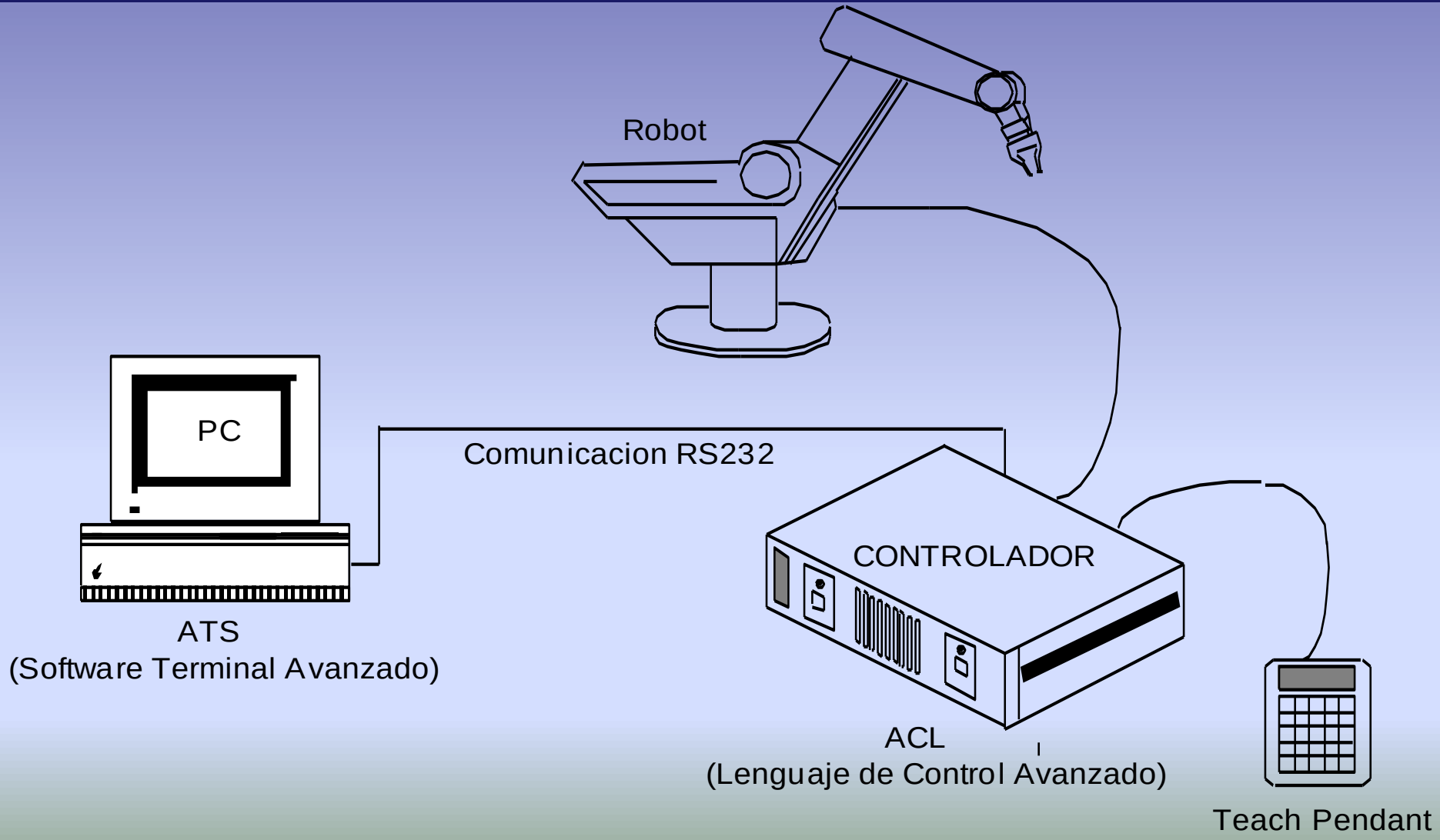
Programación de Robots

Ing. Eddie Sobrado M.

Componentes de los Robots



Componentes de los Robots

Brazo manipulador

Teach Pendant

Controlador

Planificador de Tareas y Trayectoria de Robot

PROGRAMA

- Planificador de tareas
- Planificador de trayectorias

GENERADOR DE TRAYECTORIAS

Algoritmo de Control

Sensores

Actuadores

PROGRAMA DE USUARIO

Datos del programa usuario (variables, posiciones, parámetros de trayectoria, etc.), Ejemplo:

- Pintar coche de superficie S
- Recta de A-B, arco B-C-D, etc.

Interprete de programa

Generador trayectoria (interpola, muestrea, etc.). Ejemplo:

- Puntos XYZ(KT_m) y calcular q_i (KT_m)

Programa de control

Calcula voltajes, lee encoders, etc. Ejemplo:

Converger q_{ireal} (KT_m)-> q_i (KT_m)

SO TIEMPO REAL
multitarea
Datos internos

Esquema de bloques de célula de robots

Sensores y Actuadores

CPU 1 de brazos

CPU 2 de brazos

CPU N de brazos

Controlador central

- Programa de usuario
- Interprete de programa
- Generador de trayectoria (interpola,muestrea,etc)
- Control : calcula los voltajes
- SO. Tiempo Real

Puerto de Comunicación

Monitor central (célula)

- Sincronizar y depurar cadena
- Monitorizar cadena

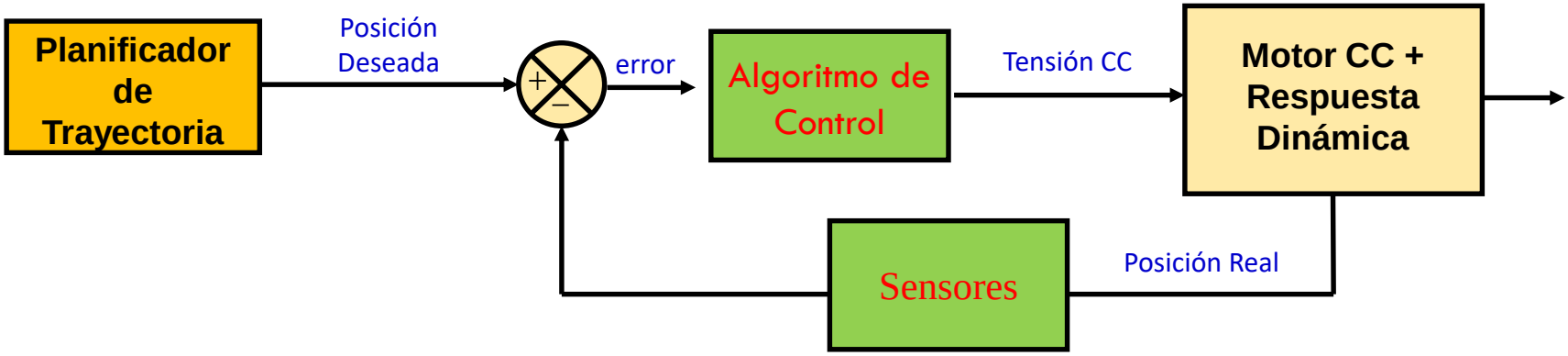
RED

PC workstation

- Escribir, simular, depurar programas
- Enviar comando (modo directo)
- Monitorizar robot

Control de Movimienos

- Generación de movimiento: mediante aplicación de fuerzas lineales o pares (rotacionales) en las articulaciones.
- Control de movimiento: calculo de los pares necesarios para conseguir posiciones y velocidades requeridas



- Características adicionales: compensación de errores y perturbaciones

Especificaciones Técnicas

ITEM	ESPECIFICACIÓN
Tipo de control	Stand Alone Tiempo Real Multitarea PID PWM Máximo 12
Numero de servo ejes	12 ejes máximo
Grupos de control	12 ejes pueden ser divididos en 3 grupos: Grupo A Grupo B Grupo C Cada grupo tiene un control independiente

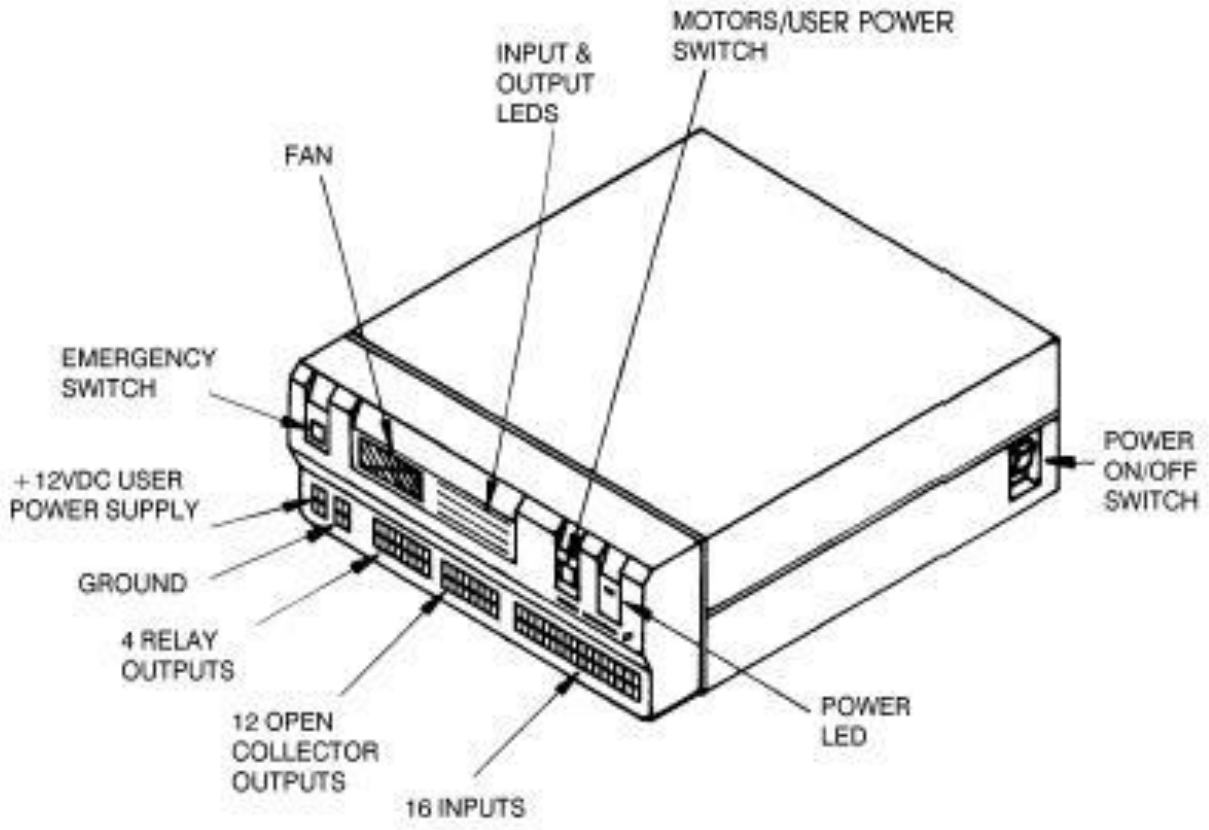
Especificaciones Técnicas

ITEM	ESPECIFICACIÓN
Tarjetas Driver de los ejes	Driver Puente H PWM 33Khz, 7ª 33-42v (dependiendo del voltaje de entrada y carga) con taco Feedback
Ruta de Control	CP (Ruta continua) Joint Lineal Circular Spline
Trayectoria de control	Parabola Sinusoide
Peso	36Kg
Relación de Engranajes	Motor 1,2,3 127:1 Motor 4,5 65:1 Motor 6 19:1

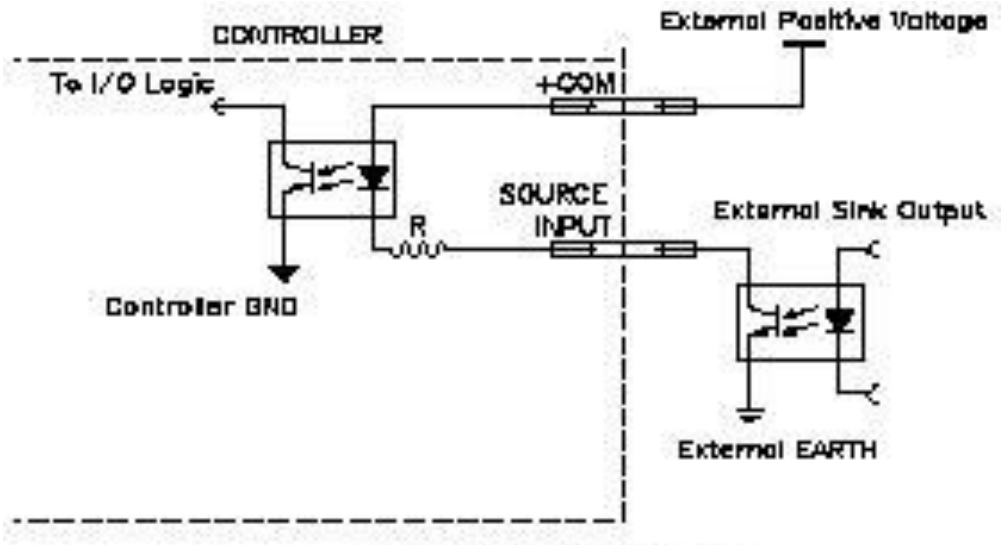
Especificaciones Técnicas

ITEM	ESPECIFICACIÓN
CPU	Motorola 68020 Motorola FPU MC68881
EPROM	512 KB
RAM CMOS	512 KB expandible a 2MB Aprox. 150KB para el sistema Aprox. 350KB disponible para el usuario
Comunicación	2 canales RS232 expandible a 10 Puerto paralelo bidireccional
Entradas/Salidas	16 Entradas 16 Salidas
Bus Interno	32 bits de datos, 24 bits de direcciones
Multitarea	En ejecución simultanea máxima: 40 tareas de usuario
Sistemas de Posicionamiento	Encoder ópticos incrementales con indicador de pulso
Sistemas de Coordenadas	Coordenadas XYZ Coordenadas Joints

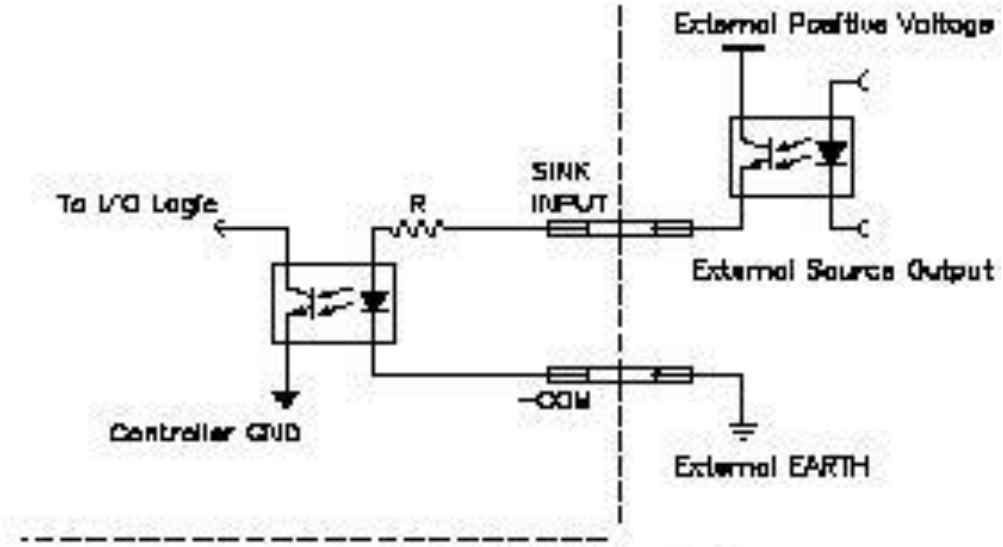
Controlador



Circuitos de Entrada: Alimentación externa

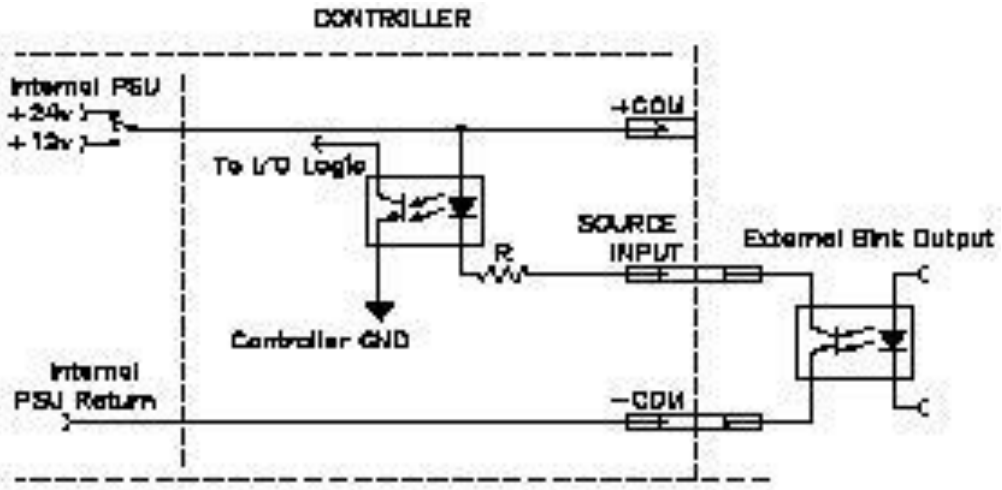


Input Circuit (Source) - External PSU

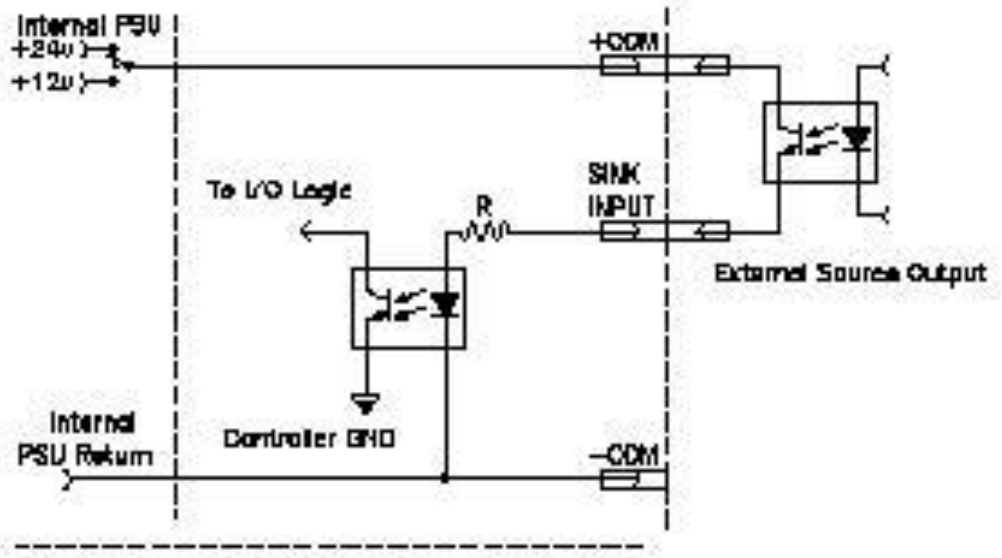


Input Circuit (Sink) - External PSU

Circuitos de Entrada: Alimentación interna

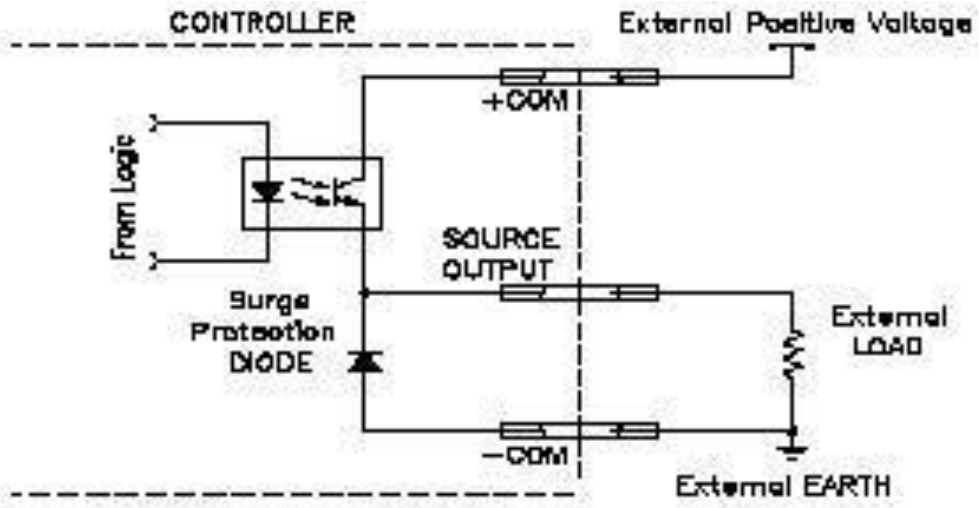


Input Circuit (Source) - Internal PSU

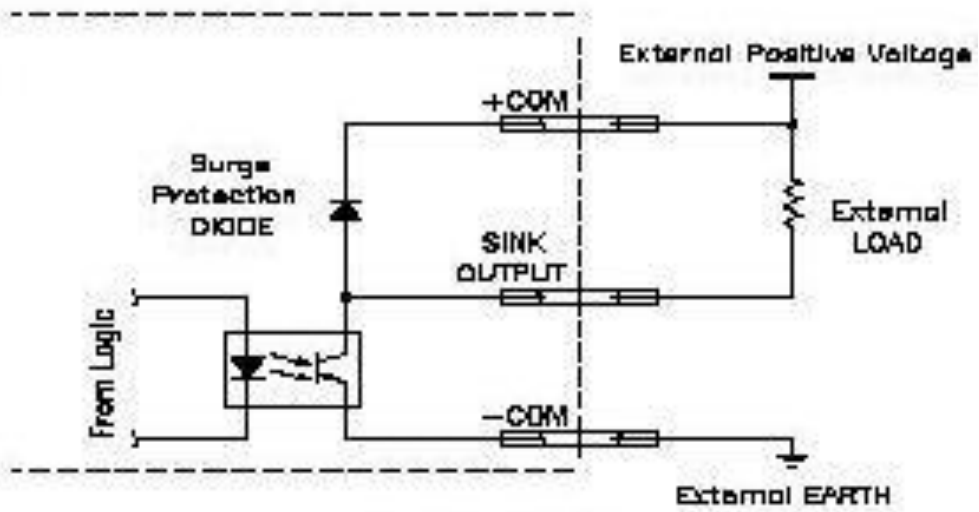


Input Circuit (Sink) - Internal PSU

Circuitos de Salida: Alimentación externa

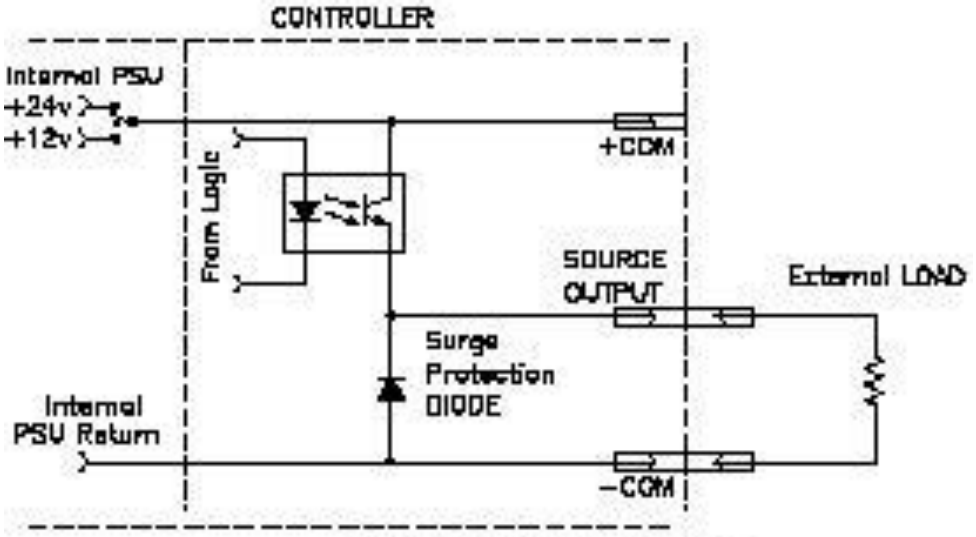


Output Circuit (Source) - External PSU

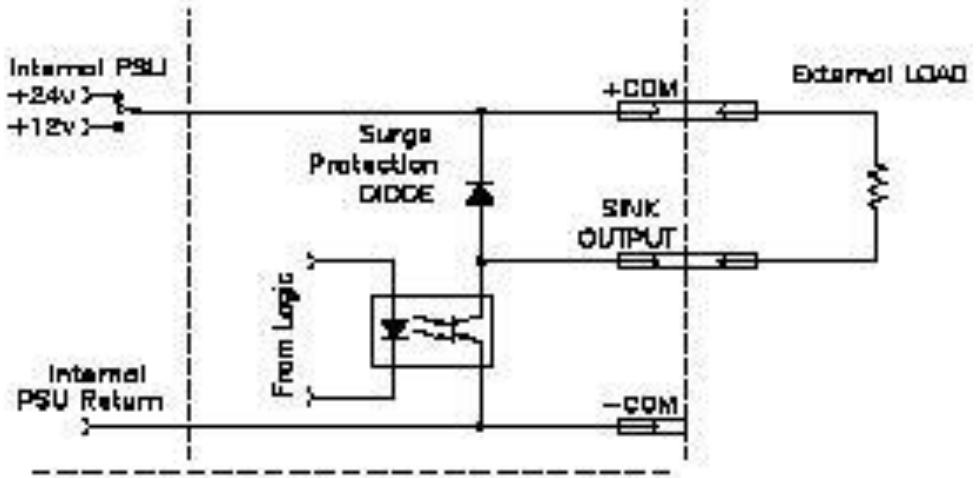


Output Circuit (Sink) - External PSU

Circuitos de Salida: Alimentación interna



Output Circuit (Source) - Internal PSU



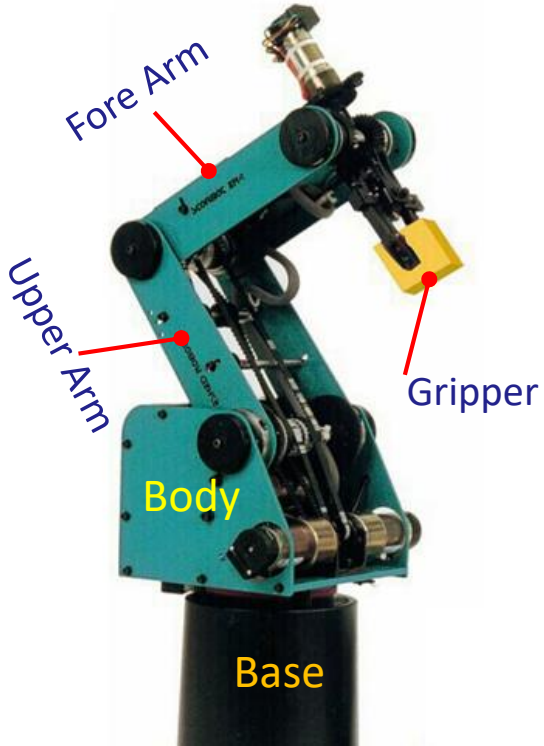
Output Circuit (Sink) - Internal PSU

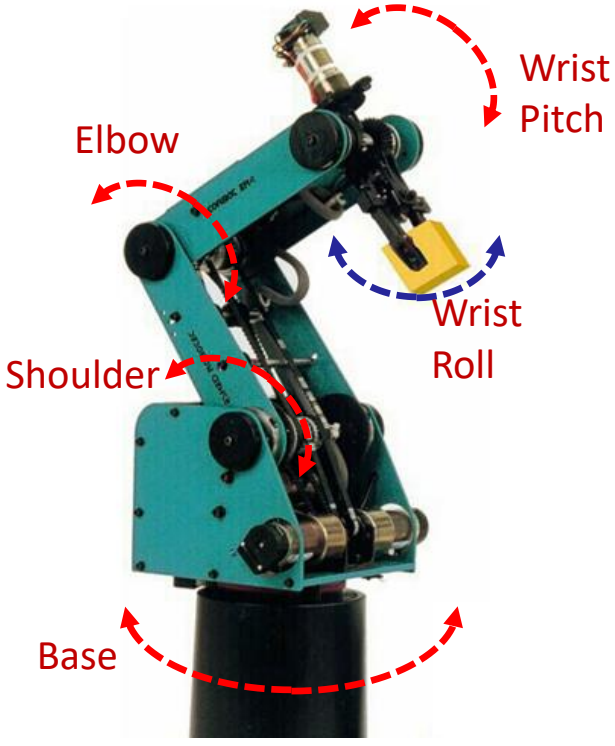
SCORBOT ER-V+

- Es un robot vertical articulado con cinco juntas rotativas.
- Con el agregado de la mordaza, el robot posee seis grados de libertad.
- Los movimientos de las juntas están descriptas en la siguiente tabla:

SCORBOT ER-V+

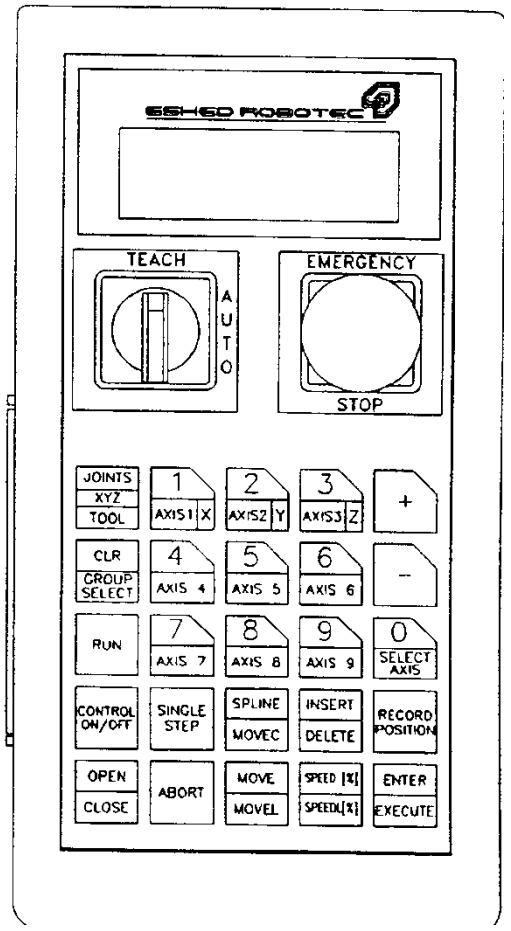
Numero de ejes	Nombre de la junta	Acción
1	Base	Gira el cuerpo (body)
2	Shoulder	Sube baja el brazo superior (uper arm)
3	Elbow	Sube o baja el antebrazo (fore arm)
4	Wrist pitch	Sube o baja la pinza (gripper)
5	Wrist Roll	Gira la pinza (gripper)





SCORBOT ER IX: Teach Pendant

- Tiene una llave selectora para elegir el modo de mando entre *auto* o *manual*. La posición auto permite operar al robot desde la PC, la posición manual deshabilita el mando desde la computadora y habilita el teach pendant.
- Posee también un pulsador de parada de emergencia con enclavamiento que desactiva los motores del brazo mientras se encuentra accionado y un botón lateral que también funciona como parada de emergencia cuando el sistema se encuentra en modo manual.



SCORBOT ER-V+: Controlador

- El controlador cuenta además con 16 entradas (destinadas a recibir señales de dispositivos externos que interactúen con el robot, como sensores, pulsadores, etc.)
- 16 salidas (destinadas a transmitir señales hacia los dispositivos externos) de las cuales 4 son a relay y las restantes son de transistor.
- En el controlador hay un botón de parada de emergencia,

SCORBOT ER-V+: Teach Pendant

- El teach pendant es un dispositivo opcional, es una terminal de mano, usado para controlar el robot y los periféricos conectados al controlador.
- Es practico para el movimiento de ejes, grabado de posiciones, direccionamiento de los ejes a posiciones grabadas y ejecución de programas.
- Trabaja tanto en coordenadas joint como en cartesianas.

SCORBOT ER IX

- El **scorbot – ER IX** es un robot vertical articulado con cinco juntas rotativas.
- Con el agregado de la mordaza y el riel, el robot posee siete grados de libertad.

SCORBOT ER IX

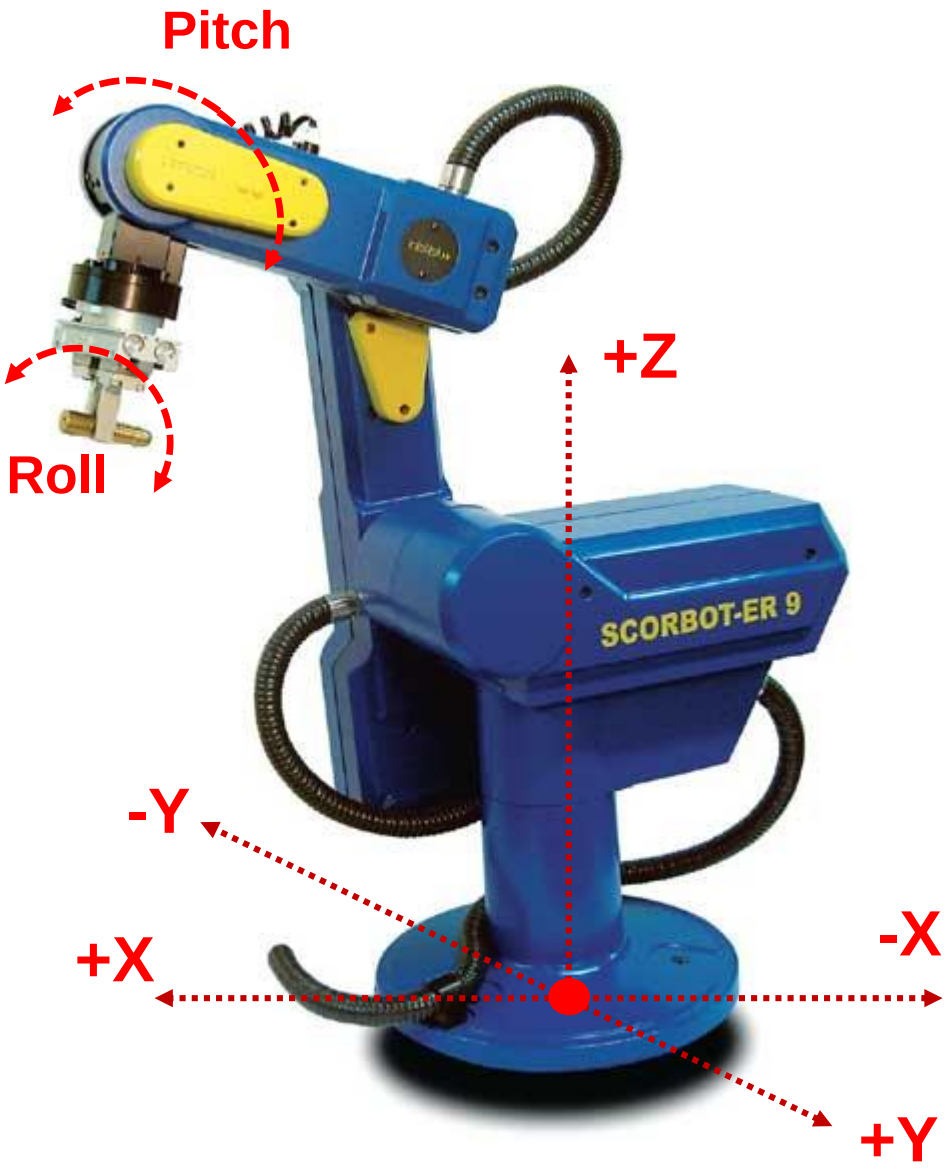
- El brazo del robot es articulado verticalmente con transmisión armónica en las cinco articulaciones.
- El sistema de retroalimentación utiliza encoders ópticos para alta repetibilidad del robot.
- SCORBOT-ER IX posee estructura mecánica fuerte, que consigue alta precisión y gran velocidad. La carga máxima es de 2kg y posee trayectoria continua completa.

SCORBOT ER IX

- El controlador SCORBOT-ER IX tiene dos canales de comunicación RS232 (ampliable a 10 canales) y un puerto paralelo bidireccional. Su CPU 68020 Motorola permite multitareas de hasta 20 programas. El co-procesador matemático Motorola MC68881 permite desplazamientos más rápidos. El controlador SCORBOT-ER IX está equipado con 6 servo ejes DC y total accesibilidad a los parámetros de control, y puede programarse desde 2 lenguajes de programación ACL y Scorbase.

SCORBOT ER IX: Sistemas de Coordenadas

- **Coordenadas joint:** especifican la posición de los ejes robóticos en unidades de encoders, ya que estos proporcionan un número de señales proporcional al movimiento de los ejes.
- **Coordenadas Cartesianas:** es un sistema geométrico para especificar la posición del punto central de la herramienta del robot TCP definiendo la distancia en unidades lineales (décimas de milímetros) desde el origen situado en la base



Lenguaje de Control Avanzado (ACL)

- Es un avanzado lenguaje de programación de robots, el mismo incluye:
  - ✓ Control directo de los ejes robóticos
  - ✓ Programación por el usuario del sistema robótico
  - ✓ Control de entradas/salidas
  - ✓ Ejecución de programas simultanea, sincronizada e interactiva; soporte multi-tasking completo.
  - ✓ Fácil gestionamiento de archivos

Características Básicas de ACL

- Dos modos de trabajo:
  - ✓ Direct
  - ✓ Edit
- Grupo A son los ejes.
- Grupo B para accesorios.
- Grupo C para un eje concreto.
- Posiciones globales
- Variables globales/privadas/sistema

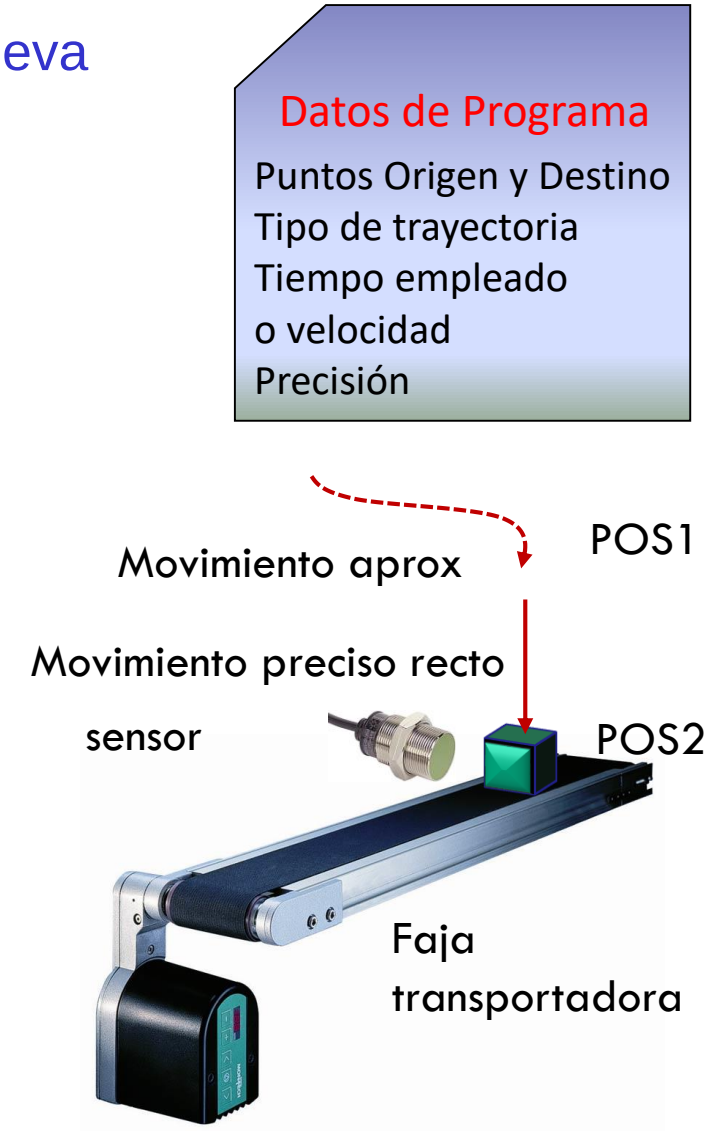
ATS (Advanced Terminal Software)

- Es la interface para el ACL. Este software es un emulador de terminal para el acceso al ACL desde una PC.
- El ATS permite:
  - ✓ Forma corta de configurar el controlador
  - ✓ Definición de periféricos
  - ✓ Editor de programas
  - ✓ Teclas de short-cut para entrar
  - ✓ Bakup manager

Ejemplos de Tarea

Tarea: Coger pieza de la cinta si hay una nueva

- Programa (lenguaje de programación):
  1. Mover a POS1 en coord propias
  2. Abrir pinza
  3. Espera sensor (detecta pieza)
  4. Desactivar cinta transportadora
  5. Mover a POS2 en línea recta
  6. Cerrar pinza
  7. Mover a POS1 en línea recta
  8. Activar cinta transportadora....



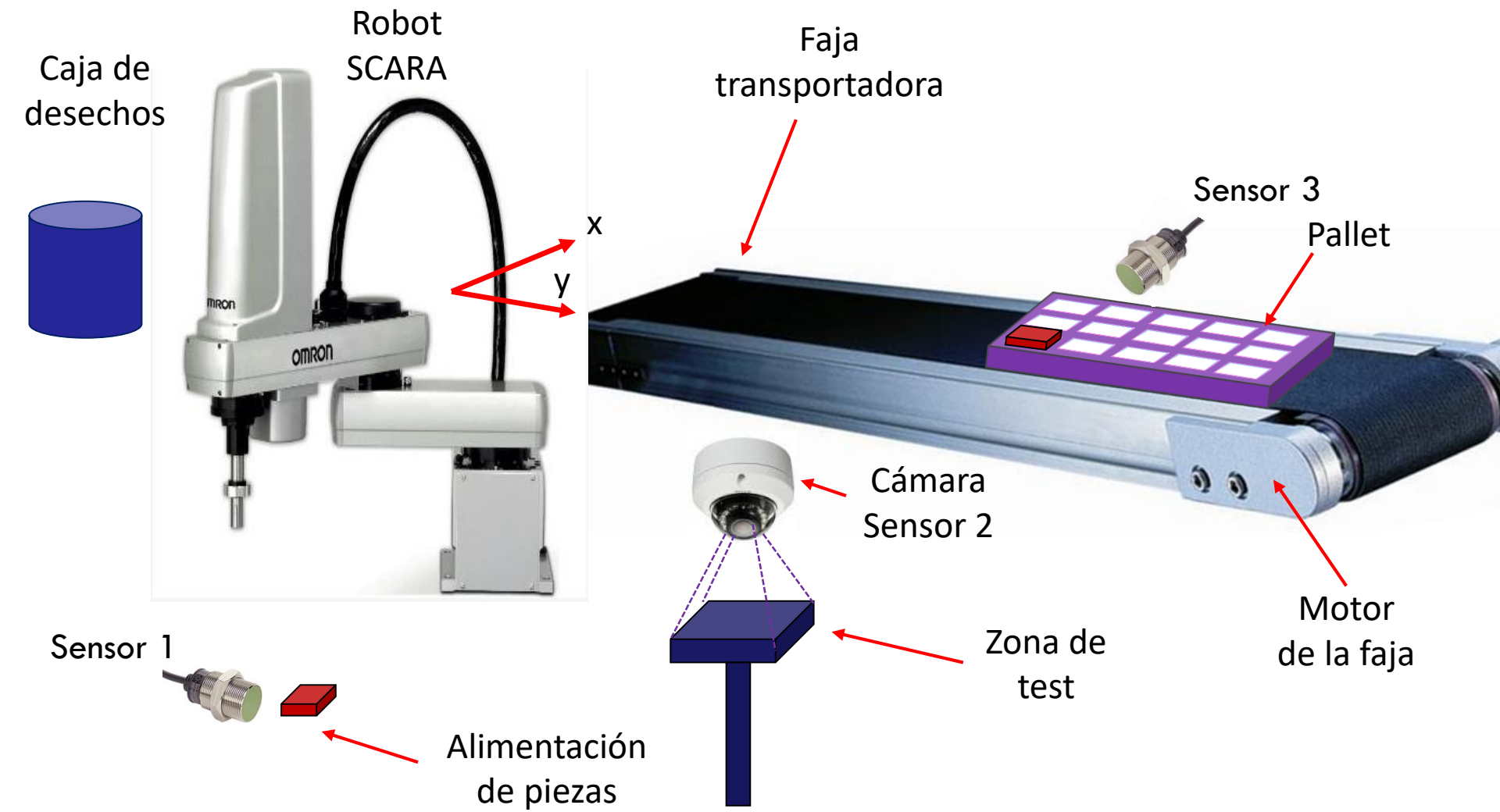
Métodos de Programación

- Depende mucho de la estructura del robot/controlador (computadores obsoletos, poca inversión en software robots)
- En la actualidad no existe lenguaje estándar.
- Cada fabricante desarrolla su método particular, válido sólo para sus propios robots
- Todos ellos tienen características y requerimientos comunes.
- Dos métodos de programación:
  - ✓ Guiado o aprendizaje: Mover físicamente el robot, registrar puntos y algunos comandos simples (botonera, paleta de programación o *teach pendant*).
  - ✓ Textual: Utilizando un lenguaje de programación.
- En la actualidad los sistemas de programación tienden a combinar estos dos métodos.

Programación Textual a nivel robot

- Ventajas (respecto a program. por guiado):
  - ✓ Estructuras de programación similares a las de los computadores (iterativas, condicionales, subrutinas).
  - ✓ Acceso a E/S con comandos (interacción entorno).
  - ✓ Sincronización fácil (órdenes “Espera”, “Mientras Entrada no activada...”, etc.).
- Inconvenientes: más complejidad.
  - ✓ Necesita work station y programador de computadores
  - ✓ Los comandos suelen depender del robot y lenguaje concretos.
  - ✓ Número elevado de instrucciones.
  - ✓ Difícil tener en cuenta accidentes, fallos, imprevistos.

Ejemplo : Paletizado

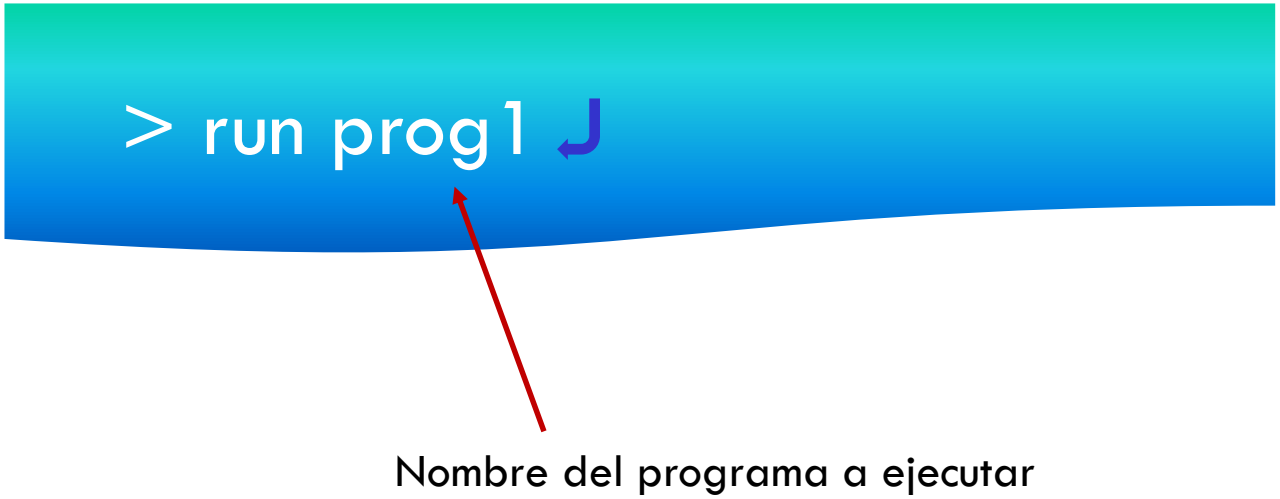


Comandos de control de programa y Tiempo Real

RUN	PRIORITY	POST
A (ABORT)	SET TIME	PEND
STOP	DELAY	
SUSPEND	WAIT	
CONTINUE	TRIGGER	

Ejecutar un Programa

- Para ejecutar un programa, escribir el siguiente comando



Controlador Multitarea

- El Controlador es un controlador en tiempo real multitarea; puede simultáneamente ejecutar y controlar 40 tareas definidas por el usuario.

```
define var
for var= 1 to 10
println 'LOOP'
wait out[4]=1
endfor
```

```
label 1
moved pos1
moved pos2
moved pos3
goto 1
endfor
```

```
define I
label 1
for i=1 to 16
    set out[1]=1
    delay 200
endfor
for i=1 to 16
    set out[1]=0
    delay 200
endfor
goto 1
```

Controlador Multitarea

- El programa PROG1 toma el robot a través de una serie de movimientos. El programa LOOP causa que texto sean desplegados en la pantalla. El programa OUT cambia las salidas del controlador a on y off.

1. Ejecución Simultanea de Procesos

- El comando RUN se puede incluir en un programa con la finalidad de iniciar la ejecución de otro programa. Cuando un programa en ejecución encuentra un comando

```
> RUN prog1 ↵
> RUN out   ↵
> RUN  Loop ↵
```

los procesos se ejecutaran concurrentemente.

- Mayor prioridad tienen precedencia; aquellos con la misma prioridad comparten la CPU del controlador por medio de un algoritmo de igual distribución.

2. Ejecución por Prioridades de Procesos

- Cuando se ejecutan varios programas concurrentemente, los que tengan mayor prioridad se ejecutaran primero
- Los programas de usuario con la misma prioridad comparten tiempo de CPU empleando un algoritmo de igual distribución
- La prioridad por defecto es 5
- El comando PRIORITY cambia tantos las prioridades por defecto así como las actuales

PRIORITY PALET 7

- Las prioridades van de 1 a 10, siendo 10 la mas alta

3. Sincronizando un Proceso por I/O

- El comando TRIGGER se puede emplear para ejecutar otro programa cuando una entrada o salida cambia a ON u OFF.
- Sin embargo; activará el programa solamente una vez, prescindiendo de cambios subsiguientes en el estado I/O.
- Ejemplo

TRIGGER DRILL BY IN 15 1

Comienza DRILL cuando se activa la entrada 15

TRIGGER TAKE BY OUT 8

Comienza TAKE cuando la salida cambia de estado

4. Suspender un Proceso

- Escriba:

SUSPEND PROG1

- El robot completará el movimiento actual y luego se detendrá. El programa prog1 esta suspendido.
- Escriba:

CONTINUE PROG1

- El comando CONTINUE causa que el robot continúe moviéndose desde el punto donde fue detenido por el comando SUSPEND.

5. Eliminar la Ejecución de un Proceso

- Use el comando STOP.
- El comando STOP abortará un programa solamente después que todos los comandos de movimientos de los ejes los cuales han sido enviados al controlador (movement buffer) hayan terminado su ejecución.
- Escriba:

  STOP PROG1

  Elimina solamente el programa PROG1

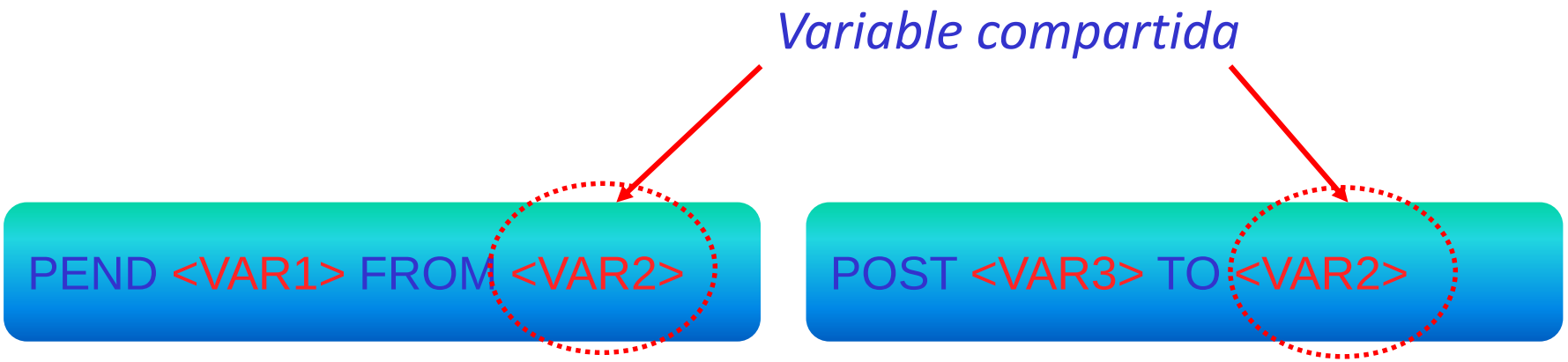
- Escriba:

  STOP

  Abortará todos los programas en ejecución

.

6. Sincronización de Procesos



- Cuando se ejecuta una instrucción:

  PEND <VAR1> ***FROM*** <VAR2>

  pueden ocurrir dos situaciones:

a. Si <var2> = 0, se suspende la ejecución del proceso hasta que se ejecute la instrucción: ***POST*** <VAR3> ***TO*** <VAR2> en otro proceso que consiste en poner un valor distinto de cero en <var2>
b. Si <var2> ≠ 0, entonces
   - <var1> ≠<var2>
   - <var2> = 0

Ejemplo 1: Sincronización de Procesos

- GLOBAL BLOQ ; Variable para bloquear y desbloquear

```
PROGRAM UNO

set bloq=0
pend var1 from bloq
run pieza
end
```

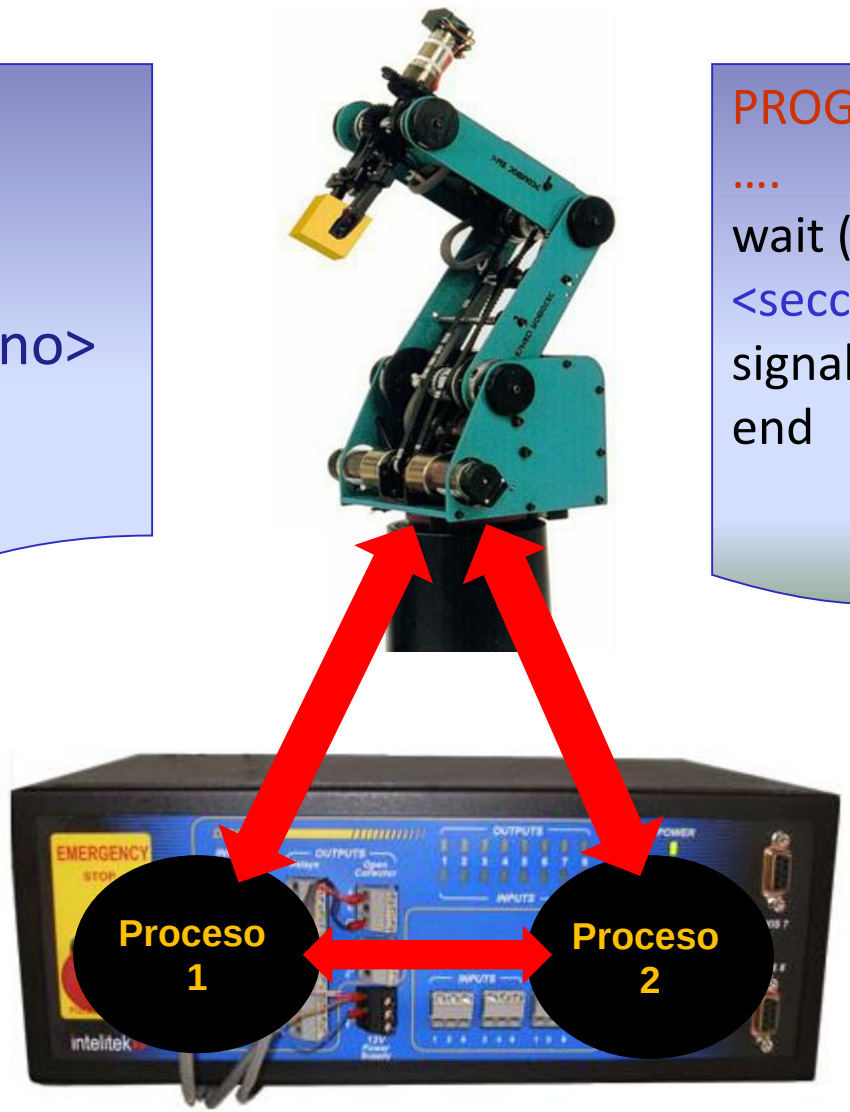
```
PROGRAM DOS

post 1 to bloq
end
```

- La ejecución del programa UNO se suspende hasta que se ejecute el programa DOS, asignando el valor 1 a BLOQ

Ejemplo 2: compartir el recurso del Robot

PROGRAM UNO

....
wait (bloq)
<seccion critica uno>
signal (bloq)
end



Controlador

PROGRAM DOS

....
wait (bloq)
<seccion critica dos>
signal (bloq)
end

Ejemplo 2: compartir el recurso del Robot

```
PROGRAM UNO
....
define var1
label 1
pend var1 from bloq
   <seccion critica uno>
post  1  to bloq
goto 1
end
```

Proceso 1

```
PROGRAM DOS
....
define var3
label 2
pend var3 from bloq
   <seccion critica dos>
post  1  to bloq
goto 2
end
```

Proceso 2