|  |
| --- |
| SFC Template |
| Kollmorgen Automation Suite - KAS |
| Escrito por: Souza, Leonardo |
| 09/2018 – rev1.0 |

**Sumário**

[1.0 Introdução 3](#_Toc20229705)

[2.0 Especificação 4](#_Toc20229706)

[2.1 Rede EtherCAT 4](#_Toc20229707)

[2.2 Entradas e Saídas 4](#_Toc20229708)

[2.2.1 Entradas Digitais 4](#_Toc20229709)

[2.2.2 Saídas Digitais 5](#_Toc20229710)

[2.2.3 Entradas Analógicas 5](#_Toc20229711)

[2.2.4 Saídas Analógicas 5](#_Toc20229712)

[3.0 Como o Standard SFC Template funciona? 6](#_Toc20229713)

[3.1 Dicionário 7](#_Toc20229714)

[3.1.1 Nome das Variáveis 8](#_Toc20229715)

[3.1.2 Entradas e saídas físicas 8](#_Toc20229716)

[3.1.3 Nome dos Subprogramas 9](#_Toc20229717)

[3.1.4 Uso da estrutura em conexão com subprogramas 9](#_Toc20229718)

[3.2 Main – Programa SFC – Rotina da Máquina de Estado 10](#_Toc20229719)

[3.2.1 Boot – Child SFC - Perfis de inicialização e movimento de rede 11](#_Toc20229720)

[3.2.2 Home – Child SFC – Rotina de referência de eixos 12](#_Toc20229721)

[3.2.3 Manual – Child SFC – Modo Manual 13](#_Toc20229722)

[3.2.4 Automatic – Child SFC – Modo Automático 15](#_Toc20229723)

[3.2.4.1 F1\_TrapzoidMove – Child SFC – Movimento trapezoidal ponto a ponto 16](#_Toc20229724)

[3.2.4.2 F2\_ CamGear – Child SFC – Movimento por Came ou Gearing 18](#_Toc20229725)

[3.2.4.3 F3\_CAMonFly – Child SFC - Camming on the Fly 19](#_Toc20229726)

[4.0 Como dimensionar o Template SFC padrão para uma máquina de 7 eixos? 21](#_Toc20229727)

[4.1 Repartição do Projeto 22](#_Toc20229728)

[4.2 Rotina Main 23](#_Toc20229729)

[4.3 Rotina de Homing 24](#_Toc20229730)

[4.4 Modo Automático 25](#_Toc20229731)

# Introdução

Se olharmos para os projetos de CLP, vemos que toda máquina lida com as mesmas funcionalidades. Geralmente, máquinas alimentadas por servoconversores precisam seguir uma sequência de etapas durante seu desenvolvimento:

1. *Inicie um protocolo de rede (se disponível);*
2. *Ligue todos os drives;*
3. *Eleger um método de referência para cada eixo;*
4. *Desenvolver um modo manual, permitindo ao operador movimentar um eixo e alterar as posições de configuração;*
5. *Finalmente, desenvolvimento de um Modo Automático, que executa as funcionalidades desejadas da máquina;*

Todas as etapas descritas acima são as partes principais do projeto e, contidas no programa Main, estabelecem um caminho para o código da máquina a ser desenvolvido. Além dessas etapas, o programa requer alguns Programas Auxiliares para lidar com Status, Falhas, I/Os, Receita e/ou qualquer rotina especial.

Este Template SFC padrão propõe uma divisão das funcionalidades conhecidas em módulos, organizando-as usando a alta potência expressiva fornecida pelo KAS usando a linguagem SFC. O objetivo é fornecer uma estrutura capaz de controlar máquinas de diferentes tipos e níveis de complexidade, usando os mesmos Programas Auxiliares e a estrutura SFC.

**Sequential Function Chart (SFC)**

A linguagem SFC é um diagrama de estado. As etapas gráficas são usadas para representar estados estáveis e as transições descrevem as condições e os eventos que levam a uma mudança de estado. O uso do SFC simplifica muito a programação de operações sequenciais, pois economiza muitas variáveis e testes apenas para manter o contexto do programa.

*DICA: é importante conhecer os Fundamentos da SFC para entender o Template padrão em SFC. Um documento explicando como funciona pode ser encontrado na Ajuda do KAS, seção:*

[*Technical References*](file:///C:/Program%20Files%20(x86)/Kollmorgen/Kollmorgen%20Automation%20Suite/Help/Content/11.TechRefs/Technical%20References.htm) *>* [*Programming Languages*](file:///C:/Program%20Files%20(x86)/Kollmorgen/Kollmorgen%20Automation%20Suite/Help/Content/Straton/lgoverview.htm) *> Sequential Function Chart*

# Especificação

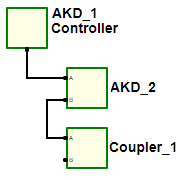
Este Template usa o KAS (Kollmorgen Automation Suite) para criar uma solução completa de automação, fornecendo suporte ao Motion Control, PLC e IHM. O programa utiliza dois idiomas IEC-61131: SFC como um programa de nível superior, construindo a máquina de estado e o ST para a codificação dentro dos qualificadores de etapa e em Programas Auxiliares.

O PLCOpen foi escolhido como mecanismo de movimento para a aplicação, composto por 2 eixos como tipo Servo e 1 eixo como mestre virtual. Todos eles foram configurados na proporção de 1: 1, com 360 graus por rotação e sem posição de rollover.

Usando o Demo KAS 2 Eixos como hardware básico de desenvolvimento, este Template é totalmente adaptado para executar com a configuração do Demo ou usando o KAS Simulator.

## Rede EtherCAT

Este Template usa o KAS (Kollmorgen Automation Suite) para criar uma solução completa de automação, fornecendo



*Figura 1. Template da rede EtherCAT.*

## Entradas e Saídas

## Entradas Digitais

AKD01 X7:10 – DIN1 – Limit Switch 1

AKD01 X7:09 – DIN2 – Free

AKD01 X7:04 – DIN3 – Free

AKD01 X7:03 – DIN4 – Free

AKD01 X8:06 – DIN5 – Free

AKD01 X7:05 – DIN6 – Free

AKD01 X7:02 – DIN7 – Free

AKD01 X35:02 – DIN21 – Free

AKD01 X35:03 – DIN22 – Free

AKD01 X35:04 – DIN23 – Free

AKD01 X36:02 – DIN24 – Free

AKD01 X36:03 – DIN25 – Free

AKD01 X36:04 – *DIN26* – Free

AKD02 X7:10 – DIN1 – Limit Switch 2

AKD02 X7:09 – DIN2 – Free

AKD02 X7:04 – DIN3 – Free

AKD02 X7:03 – DIN4 – Free

AKD02 X8:06 – DIN5 – Free

AKD02 X7:05 – DIN6 – Free

AKD02 X7:02 – DIN7 – Free

## Saídas Digitais

AKD01 X7:7 – DOUT1 – Free

AKD01 X7:5 – DOUT2 – Free

AKD01 X35:7 – DOUT21 – Free

AKD01 X36:7 – DOUT22 – Free

AKD02 X7:7 – DOUT1 – Free

AKD02 X7:5 – DOUT2 – Free

## Entradas Analógicas

AKD01 X8:10 – AIN – Free

Auxilia

AKD02 X8:10 – AIN – Free

## Saídas Analógicas

AKD01 X8:08 – AOUT – Free

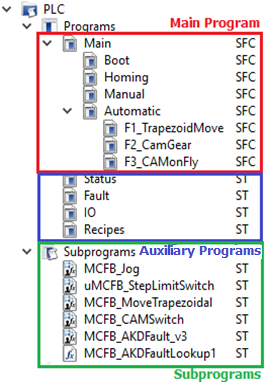
AKD02 X8:08 – AOUT – Free

# Como o Standard SFC Template funciona?

O Template SFC padrão divide o Projeto KAS em "Grupos de Funcionalidades", as rotinas relacionadas ao processo da máquina estão localizadas dentro do Programa Principal e os Programas Auxiliares são desenvolvidos usando Programas Independentes.

A maior vantagem do uso da estrutura SFC é converter o processo da máquina em um diagrama de estados. SFC é uma boa escolha porque:

* Cada estado do processo pode ser claramente mapeado para uma etapa do SFC;
* A transição entre etapas é feita usando condições, estas são facilmente identificadas no código;
* A relação entre etapas e fluxo de atividade de um estado para outro é visualizada/programada de maneira gráfica, é fácil compreender as devidas condições e intertravamentos que relacionam a evolução dos estados ativos da etapa.
* A estrutura SFC é um espelho do processo controlado e permite que o software quebre em partes pequenas. Com esse recurso, é fácil controlar o que está sendo executado em cada ciclo de verificação do controlador.
* As máquinas geralmente têm diagramas de estado semelhantes (Homing, Modo Manual, Modo Automático ...), essa característica permite a escalabilidade do Template SFC Padrão em máquinas mais complexas.

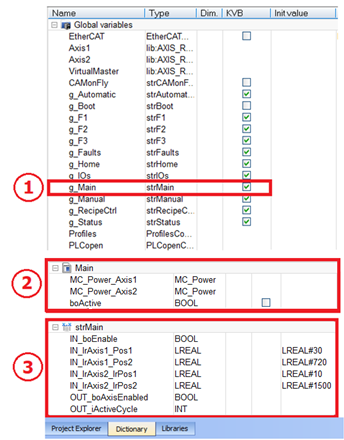
O **programa Main** é responsável por converter o diagrama de estado da máquina em software PLC. É responsável por inicializar a rede, endereçar, fornecer energia e referenciar todos os eixos, além de executar todas as funcionalidades nos Modos Manual e Automático.

**Programas auxiliares** tem como objetivo fornecer informações ao Programa Main. Eles possuem ciclos de execução estabelecidos, sem quaisquer condições de intertravamento, ou seja, continuam sempre funcionando. O template usa esses programas para ler o status da máquina, lidar com falhas e redefinir rotinas e gerenciar IOs e receitas.

Os **subprogramas** são partes de código desenvolvidas para executar uma função específica. Eles são usados dentro de programas chamados como instâncias declaradas no dicionário do projeto.

Além de toda a estrutura de Programas de PLC descrita acima, este projeto possui uma seção para configurar o Motion e o Eixo PLCOpen e gerenciar dispositivos de rede EtherCAT. Temos duas interfaces de usuário diferentes, primeiro usando painéis de controle dentro do KAS e segundo com um projeto IHM, desenvolvido usando o KVB (Kollmorgen Visualization Builder).

## Dicionário



*Figura 3. Dicionário do Projeto. 1. As principais estruturas são declaradas como variável global. 2. Principais variáveis locais. 3. Estrutura Main.*

Cada programa possui um grupo de variáveis dedicadas, relacionadas à sua funcionalidade. Essas variáveis são divididas em dois grupos:

**Variáveis Locais**

Elas podem ser acessadas apenas dentro de seu programa. Na Figura 3, o número 2 mostra variáveis locais que pertencem ao programa principal.

**Váriaveis Globais e Estruturas**

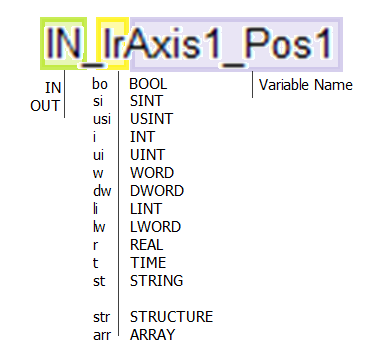
Essas variáveis são especiais, portanto podem ser acessadas em todo o projeto. Durante o desenvolvimento do template, estabelecemos que cada programa possui uma estrutura de variável com o seu nome: "str + nome do programa". A adoção dessa definição é fácil de identificar de qual programa a Variável Global veio.

Na Figura 3, o número 3 mostra a Estrutura das Variáveis que pertence ao Programa Principal e o número 1 mostra que está sendo declarada como Variável Global.

*DICA: Durante o desenvolvimento, é recomendável criar todas as variáveis do programa como Local e alterá-lo para Global somente se necessário.*

## Nome das Variáveis

Para permitir a identificação se a variável representar uma Entrada ou Saída e seu Tipo, o nome da variável é dividido em 3 partes:



´ - Indique se a variável é uma entrada ou saída;

(**Somente para variáveis globais.** Variáveis locais não têm essa parte);

´ - Indica o tipo de dados da variável;

´ - Nome da variável;

## Entradas e saídas físicas

****

## Nome dos Subprogramas

Algumas partes dos códigos podem ser reutilizadas várias vezes ou para trabalhos diferentes e, portanto, são programadas como subprogramas. Isso minimiza a quantidade de partes de código duplicadas. O subprograma usa o prefixo UDFB seguido por um underline e, em seguida, o nome individual.

e.g. UDFB\_AxisHND, UDFB\_AxisSDO

Essas estruturas podem ser instanciadas em qualquer nível da programação. Começando com g se global, caso contrário, inicie com INST, seguido de underline mais o nome individual do UDFB e a numeração, se usado várias vezes.

e.g. Inst\_AxisHND1, g\_Inst\_AxisSDO2

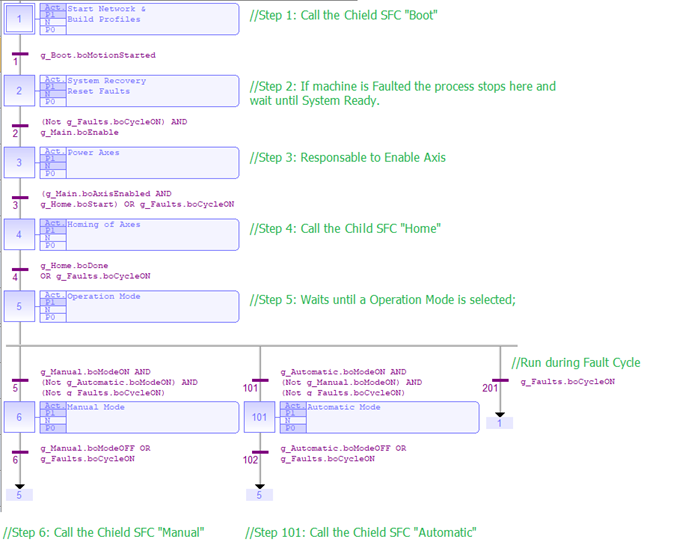
## Uso da estrutura em conexão com subprogramas

A nomeação de estrutura usa str como tipo seguido pelo subprograma que o está usando. Se vários subprogramas estiverem usando a mesma estrutura, verifique se os subprogramas têm um nome comum iniciado e use-o como o nome da estrutura.

e.g. strAxisHND, strAxis

## Main – Programa SFC – Rotina da Máquina de Estado

A máquina de estado e todas as funcionalidades relacionadas ao processo de máquina são desenvolvidas dentro do Programa Main, o intertravamento entre cada estado é feito usando a Estrutura SFC. Chield



*Figura 4. Programa Main: SFC estrutura é usada para construir a máquina de estado.*

Enquanto o controlador PDMM estiver em execução, o programa principal estará ativo. Indica qual estado de operação está ativo. Esse recurso facilita o processo de depuração, enquanto você está conectado ao controlador é claro para descobrir qual parte do código está em execução e onde ele para.

O conceito de organização do programa nos Programas SFC Principal e Infantil também ajuda a organizar o processo de desenvolvimento, deixando claro ao programador todas as etapas que ele precisa executar para concluir o projeto. Este método de divisão pode ser igual a uma grande variedade de equipamentos, sendo fácil criar um código para máquinas mais complexas usando essa mesma estrutura.

**Variável Global – Main – SFC Programa**

*g\_Main : strMain ; // Estrutura variável de chamada do principal como uma variável global;*

**Variáveis Locais - Main - SFC Programa**

*MC\_Power\_Axis1 : MC\_Power ; //MC\_Power para habilitar o eixo 1;*

*MC\_Power\_Axis2 : MC\_Power ; //MC\_Power para habilitar o eixo 2;*

**Estrutura de Variáveis – Main - SFC Programa**

*boEnable : BOOL ; //Chamada de rotina para habilitar os eixos.*

*boAxisEnabled : BOOL ; //* *Indica que todos os eixos estão habilitados;*

*iActiveCycle : INT ; //* *Indica o ciclo ativo do programa principal;*

*Axis1\_lrPos1 : LREAL := LREAL#30 ; //Axis 1: Setpoint de posição 1;*

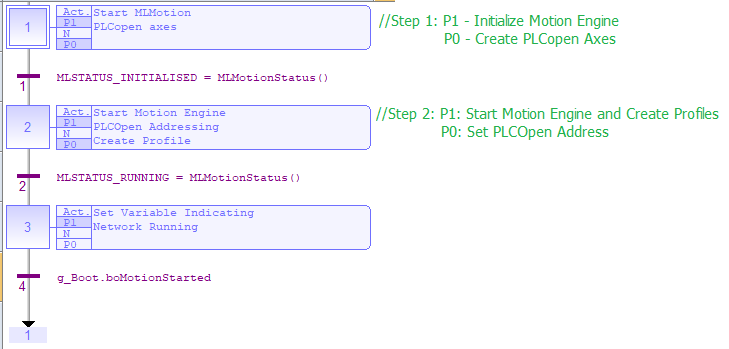
*Axis1\_lrPos2 : LREAL := LREAL #720 ; //Axis 1: Setpoint de posição 2*

*Axis2\_lrPos1 : LREAL := LREAL #10 ; //Axis 2: Setpoint de posição 1;*

*Axis2\_lrPos2 : LREAL := LREAL #1500 ; //Axis 2: Setpoint de posição 2;*

## Boot – Child SFC - Perfis de inicialização e movimento de rede

A inicialização do programa Child SFC é chamada pela 1ª etapa do programa principal. É responsável por inicializar a rede EtherCAT e o Motion Engine, definir endereços de PLCOpen e criar perfis de movimento. Ele define uma variável na Etapa 3, notando que a Inicialização está concluída.



*Figure 5. Boot: Programa SFC filho que inicia a rede da máquina.*

**Variável Global – Boot - Child Programa SFC**

*g\_Boot : strBoot ;* *//Chamada de estrutura pelo Boot como uma variável Global;*

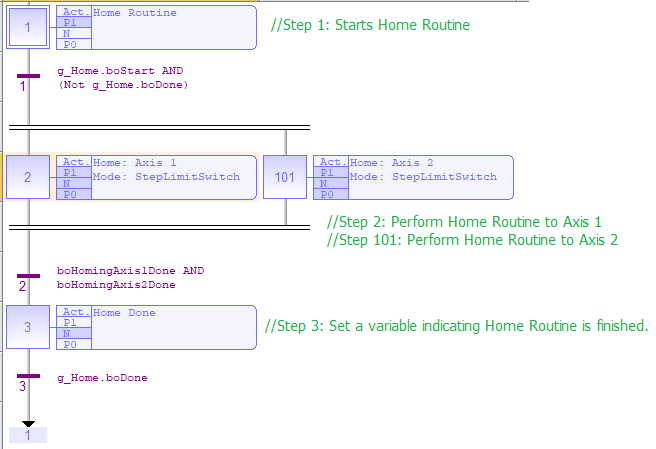
**Estrutura de Variáveis – Boot - Child Programa SFC**

*boMotionStarted : BOOL ; //Indica que a rotina Boot terminou;*

## Home – Child SFC – Rotina de referência de eixos

Todos os eixos acionados por servo precisam passar por um processo de referência durante sua inicialização. No Template, todas as rotinas de referência são centralizadas no SFC Home, que faz referência aos eixos 1 e 2, usando o bloco de função MCFB\_StepLimitSwitch.

Os dois eixos se movem até que a entrada digital mapeada no bloco de funções seja acionada, de modo que a posição de realimentação dos eixos seja zerada e então eles se movam para a respectiva Posição de Ajuste 1 de cada eixo.



*Figure 6. Home: SFC chield que executa a Rotina Doméstica nos eixos;*

**Variáveis Globais – Home – Child Programa SFC**

*g\_Home : strHome ; //Chamada de estrutura pelo Home como variável Global;*

**Variáveis Locais - Home - Child Programa SFC**

*boActive : BOOL ; //Indica que a rotina Home está ativa;*

*boHomingAxis1Done : BOOL ; // Indica que a ref. do eixo 1 está finalizada;*

*boHomingAxis2Done : BOOL ; // Indica que a ref. do eixo 2 está finalizada;*

*MC\_Halt\_Axis1 : MC\_Halt ; //* *Bloco de funções para desacelerar o eixo 1 para zero no início;*

*MC\_Halt\_Axis2 : MC\_Halt ; //* *Bloco de funções para desacelerar o eixo 2 para zero no início;*

*MCFB\_LimitSwitchRef\_Axis1 : uMCFB\_StepLimitSwitch; //Ref do eixo 1 para o fim de curso– AKD1\_IN1;*

*MCFB\_LimitSwitchRef\_Axis2 : uMCFB\_StepLimitSwitch; // Ref do eixo 2 para o fim de curso – AKD2\_IN1;*

*MC\_MoveAbsolute\_Axis1 : MC\_MoveAbsolute ; //* *Executa um movimento absoluto no eixo 1 para a posição 1;*

*MC\_MoveAbsolute\_Axis2 : MC\_MoveAbsolute ; // Executa um movimento absoluto no eixo 2 para a posição 1;*

**Estrutura de Variáveis – Home - Child Programa SFC**

*boStart : BOOL ; //Chama de rotina para referenciar os eixos;*

*boDone : BOOL ; //Indica que a rotina Home está finalizada para todos os eixos;*

*boAxis1Recovery : BOOL ; // A variável executa a rotina home pela segunda vez no eixo 1;*

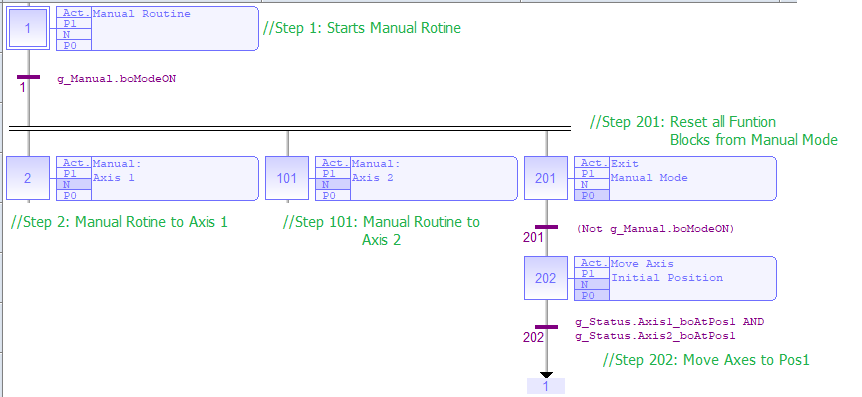
*boAxis2Recovery : BOOL ; //A variável executa a rotina home pela segunda vez no eixo 2;*

## Manual – Child SFC – Modo Manual

O Modo Manual fica disponível após a conclusão do Processo de Referência, estando em acordo com a máquina de estado do Programa Principal. Este modo permite que os eixos sejam posicionados em seus respectivos pontos de ajuste (Pos1 e Pos2) e o ajuste dessas posições usando os comandos Jog. Posteriormente ao ajuste, o operador pode salvar a nova posição, atualizando os pontos de ajuste e a receita atual.

Ao sair do modo manual, ambos os eixos são posicionados em seus respectivos pontos de ajuste (Pos1).

*DICA: É importante resetar todos os blocos funcionais utilizados nas rotinas programadas dentro das estruturas SFC. Essa condição garante que possamos sair do modo manual e entrar novamente, executando todas as rotinas.*



*Figure 7. Manual: Child SFC que executa Rotina Manual nos eixos;*

**Variáveis Globais – Home – Child Programa SFC**

*g\_Manual : strManual ; //Chamada de estrutura pelo Manual como uma variável Global;*

**Variáveis Locais - Home – Child Programa SFC**

*MC\_MoveAbsolute\_Axis1\_1 : MC\_MoveAbsolute ; //Exec. um mov. Absoluto no Eixo 1 para Pos1;*

*MC\_MoveAbsolute\_Axis1\_2 : MC\_MoveAbsolute ; // Exec. um mov. Absoluto no Eixo 1 to Pos2;*

*MC\_MoveAbsolute\_Axis2\_1 : MC\_MoveAbsolute ; // Exec. um mov. Absoluto no Eixo 2 to Pos1;*

*MC\_MoveAbsolute\_Axis2\_2 : MC\_MoveAbsolute ; // Exec. um mov. Absoluto no Eixo 2 to Pos2;*

*MCFB\_Jog\_Axis1 : MCFB\_Jog ; // Exec. um jog no eixo 1;*

*MCFB\_Jog\_Axis2 : MCFB\_Jog ; //Exec. um jog no eixo 1;*

**Estrutura de Variáveis – Home - Child Programa SFC**

*boModeON : BOOL ; //Variável usada para comandar/indicar que o Modo Manual está ON;*

*boModeOFF : BOOL ; // Variável usada para comandar/indicar que o Modo Manual está OFF;*

*Axis1\_boJogMinus : BOOL ; //* *Habilita um Jog na direção Mais Eixo 1;*

*Axis1\_boJogPlus : BOOL ; //* *Habilita um Jog na direção Menos em Eixo 1;*

*Axis1\_boSavePos : BOOL ; //* *Salve o ajuste atual em Pos1 ou Pos2 e na receita atual;*

*Axis1\_boSelPos1Pos2 : BOOL ; //* *Seleciona e posiciona o Eixo1 em Pos1 ou Pos2;*

*Axis2\_boJogMinus : BOOL ; // Habilita um Jog na direção Plus no Eixo 2;*

*Axis2\_boJogPlus : BOOL ; // Habilita um Jog na direção Menos em Eixo 2;*

*Axis2\_boSavePos : BOOL ; //Salve o ajuste atual em Pos1 ou Pos2 e na receita atual;*

*Axis2\_boSelPos1Pos2 : BOOL ; //* *Seleciona e posiciona o Eixo2 em Pos1 ou Pos2;*

*Axis1\_lrJogSpeed : LREAL := LREAL#10 ; //* *Velocidade de jog para o Eixo 1;*

*Axis1\_lrJogAccDecel : LREAL := LREAL#1000 ; // Acc/Dec de Jog Linear para o Eixo 1;*

*Axis2\_lrJogSpeed : LREAL := LREAL#10 ; // Velocidade de jog para o Eixo 2;*

*Axis2\_lrJogAccDecel : LREAL := LREAL#1000 ; // Acc/Dec de Jog Linear para o Eixo 2;*

## Automatic – Child SFC – Modo Automático

O modo automático também está disponível após a conclusão da rotina home. Como outras partes do programa, ele também é orientado para agrupar o código em funcionalidades.

Três funções diferentes estão disponíveis no modo automático (F1, F2 e F3). Este programa não é responsável por executar as funcionalidades, apenas faz o intertravamento e chama um Child SFC para executá-las..

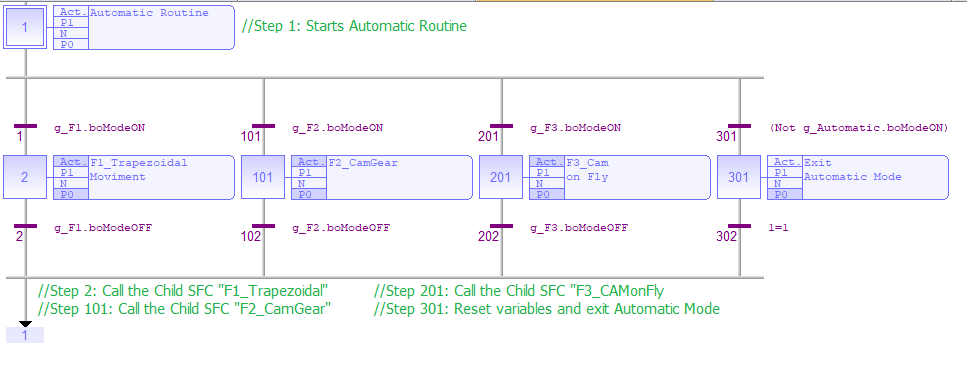
**Variáveis Globais – Automatic – Child Programa SFC**

*g\_Automatic : strAutomatic ; //Chamada de estrutura pelo Automático como variável global;*

**Estrutura de Variáveis – Automatic - Child Programa SFC**

*boModeON : BOOL ; //* *Variável usada para comandar/indicar que o Modo Automático está ON;*

*boModeOFF : BOOL ; //* *Variável usada para comandar / indicar que o Modo Automático está OFF;*



*Figura 8. Automatic: Child SFC executa rotina automática;*

## F1\_TrapzoidMove – Child SFC – Movimento trapezoidal ponto a ponto

A primeira característica executada pelo Modo Automático é responsável por mover os eixos usando um comando de posição absoluta. A distância percorrida é definida pelas posições P1 e P2.

O cálculo do movimento trapezoidal é feito pela rotina auxiliar “MCFB\_MoveTrapezoidal”, o tempo desejado para o movimento é usado como entrada e o bloco fornece velocidade e aceleração de comando para executar um movimento trapezoidal, que 1/3 do tempo é usado para acelerar, 1/3 é usado para velocidade constante e 1/3 é usado para desacelerar.

**Variáveis Global – F1\_TrapzoidMove – Child Programa SFC**

*g\_F1 : strF1; //* *Estrutura variável de chamada de F1 como uma variável global;*

**Variáveis Local – F1\_TrapzoidMove – Child Programa SFC**

*boAxis1\_P1P2 : BOOL; //* *Executa o movimento no Eixo1 de P1 a P2;*

*boAxis1\_P2P1 : BOOL; //* *Executa o movimento no Eixo 1 de P2 para P1;*

*boAxis2\_P1P2 : BOOL; //* *Executa o movimento no Eixo 2 de P1 a P2;*

*boAxis2\_P2P1 : BOOL; //* *Executa o movimento no Eixo 2 de P2 para P1;*

*lrAxis1Distance : LREAL; //* *Calcular Eixo1 Distância absoluta entre P1-P2;*

*lrAxis2Distance : LREAL; //* *Calcular Eixo2 Distância absoluta entre P1-P2;*

*MC\_Halt\_Axis1 : MC\_Halt; //* *Bloco de funções para desacelerar o Eixo 1 para zero;*

*MC\_Halt\_Axis2 : MC\_Halt; //* *Bloco de funções para desacelerar o Eixo 2 para zero;*

*MC\_MoveAbsolute\_Axis1 : MC\_MoveAbsolute; //Executa um movimento absoluto no Eixo 1*

*MC\_MoveAbsolute\_Axis2 : MC\_MoveAbsolute; //* *Executa um movimento absoluto no Eixo 2*

*MCFB\_MoveTrapezoidal\_Axis1\_P1P2 : MCFB\_MoveTrapezoidal; //Calcula Mov. Trap. do eixo 1*

*MCFB\_MoveTrapezoidal\_Axis2\_P1P2 : MCFB\_MoveTrapezoidal; // Calcula Mov. Trap. do eixo 2*

*MyBlink : blink ; //* *Trigger para* *Mover eixo 1 de P1 para P2*

*MyBlink1 : blink ; // Trigger para* *Mover eixo 2 de P1 para P2*

**Estrutura de Variáveis – F1\_TrapzoidMove – Child Programa SFC**

*boModeON : BOOL ; //* *Variável usada para comandar/indicar que F1 está ON;*

*boModeOFF : BOOL ; //* *Variável usada para comandar/indicar que F1 está OFF;*

*Axis1\_tMoveTime : TIME := T#300ms ; //* *Ponto de ajuste do tempo para o movimento Eixo1 de P1-P2;*

*Axis1\_lrVelMax : LREAL := LREAL#1000 ; //Setpoint de velocidade para o eixo 1 de P1-P2;*

*Axis1\_lrAccMax : LREAL := LREAL#10000 ; //Setpoint de Accel. para o eixo 1 mover de P1-P2;*

*Axis1\_lrRatio : LREAL := LREAL#1 ; //Eixo 1 Relação entre o físico e o PLC Open;*

*Axis1\_lrVelCmd : LREAL ; //* *Cmd de Vel. Calc. para executar o movimento de P1-P2;*

*Axis1\_lrAccDecelCmd : LREAL ; // Cmd de. Accel. para executar o movimento de P1-P2;*

*Axis1\_tErrMove : TIME ; //* *Erro entre o tempo definido e calculado para mover o eixo 1 de P1-P2;*

*Axis2\_tMoveTime : TIME := T#1s ; //Setpoint do tempo do para mover o eixo 2 de P1-P2;*

*Axis2\_lrVelMax : LREAL := LREAL#500 ; // Setpoint de velocidade para o eixo 2 de P1-P2;*

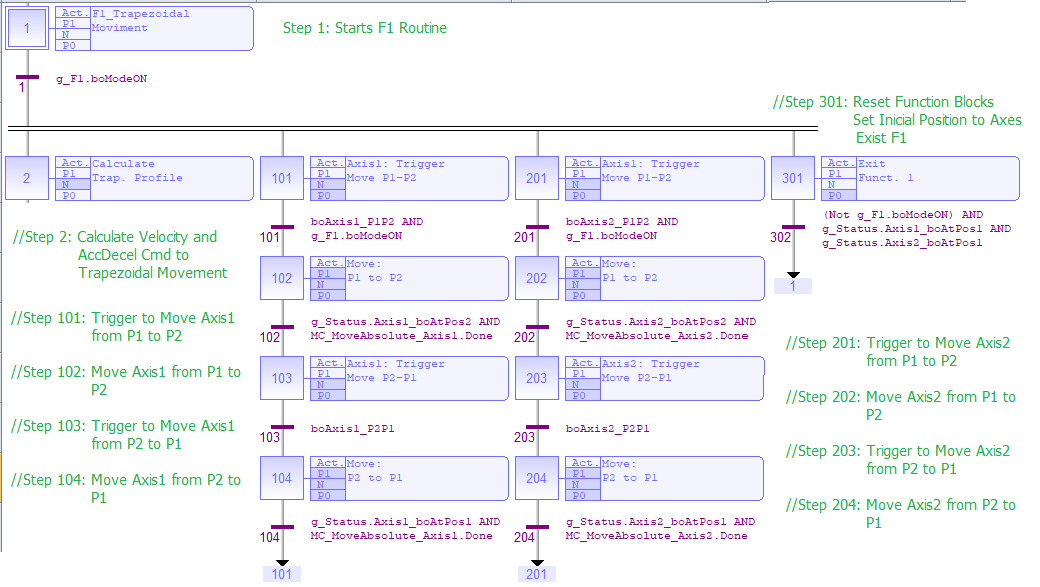
*Axis2\_lrAccMax : LREAL := LREAL#10000 ; // Setpoint de Accel. para o eixo 2 mover de P1-P2;*

*Axis2\_lrRatio : LREAL := LREAL#1 ; // Eixo 2 Relação entre o físico e o PLC Open;*

*Axis2\_lrVelCmd : LREAL ; // Cmd de Vel. Calc. para executar o movimento de P1-P2;*

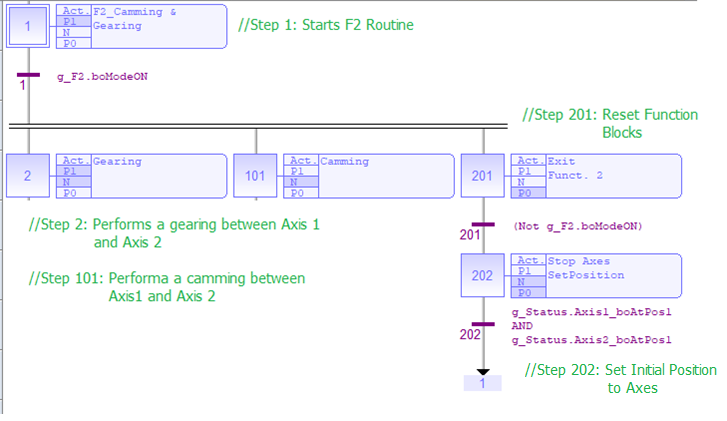
*Axis2\_lrAccDecelCmd : LREAL ; // Cmd de. Accel. para executar o movimento de P1-P2;*

*Axis2\_tErrMove : TIME ; // Erro entre o tempo definido e calculado para mover o eixo 2 de P1-P2;*



*Figura 9. F1\_TrapzoidMove: Child SFC que executa a rotina F1;*

## F2\_ CamGear – Child SFC – Movimento por Came ou Gearing



*Figura 10. F2\_CamGear: Child SFC que executa a rotina F2;*

**Variáveis Globais – Automatic – Child Programa SFC**

*g\_F2 : strF2; //Chamada da estrutura pelo F2 como uma variável global;*

**Váriaveis Locais – Automatic – Child Programa SFC**

*Inst\_MC\_SetPos\_Axis1 : MC\_SetPos ; //* *Define a posição inicial (P1) como eixo1;*

*Inst\_MC\_SetPos\_Axis2 : MC\_SetPos ; //* *Define a posição inicial (P1) como eixo 2;*

*MC\_MoveContVel\_Axis1 : MC\_MoveContVel ; //* *Mover velocidade para Axis1 Master;*

*MC\_CamTblSelect\_Profile1 : MC\_CamTblSelect ; //* *Leia e inicialize o perfil especificado*

*MC\_GearIn\_Axis2 : MC\_GearIn //* *O eixo 2 segue o eixo 1 com base em uma proporção;*

*MC\_CamIn\_Profile1 : MC\_CamIn ; //* *O eixo 2 segue o eixo 1 com base na tabela Cam;*

*MC\_Halt\_Axis1 : MC\_Halt ; //* *Bloco de funções para desacelerar o Eixo1 para zero;*

*MC\_Halt\_Axis1\_1 : MC\_Halt ; //* *Bloco de funções para desacelerar o Eixo1 para zero;*

*MC\_Halt\_Axis2 : MC\_Halt ; //* *Bloco de funções para desacelerar o Eixo 2 para zero;*

*MC\_Halt\_Axis2\_1 : MC\_Halt ; //* *Bloco de funções para desacelerar o Eixo 2 para zero;*

**Estrutura de Variáveis – Automatic - Child Programa SFC**

*boModeON : BOOL ; //* *Variável usada para comandar/indicar que F2 está ON;*

*boModeOFF : BOOL ; // Variável usada para comandar/indicar que F2 está OFF;*

*boStartMove : BOOL ; //* *Executa movimento do Eixo1 a uma velocidade especificada;*

*boEngage\_Gearing : BOOL ; //* *Ative a engrenagem entre o Eixo1 e o Eixo2, usando a engrenagem;*

*boGearingEngaged : BOOL ; //* *Indique que o Axis2 está acoplado no Axis1;*

*boEngage\_Camming : BOOL ; //* *Habilita a came entre o Eixo 1 e o Eixo 2;*

*boCammingEngaged : BOOL ; //* *Indique que o Eixo 2 está engatado no Eixo 1, usando came;*

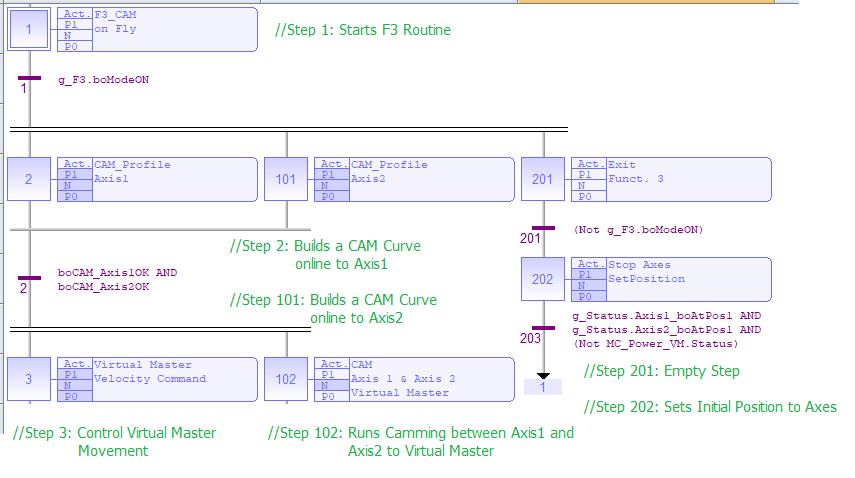
*boStopEngagement : BOOL ; //* *Desengata o Eixo 2 do Eixo 1;*

*lrMasterVel : LREAL := LREAL#1000 ; //* *Definir velocidade do comando Eixo 1 (Master);*

*diRatioNumerator : DINT := DINT#2 ; //* *Numerador da razão Eixo 1 / Eixo 2*

*diRatioDenominator : DINT := DINT#1 ; //Denominador da razão Eixo 1 / Eixo 2*

## F3\_CAMonFly – Child SFC - Camming on the Fly



*Figure 11. F3\_CAMonFly: Child SFC executa a rotina F3;*

**Variável Global – F3\_CAMonFly – Child Programa SFC**

*g\_F3 : strF3; //Chama a variável como estrutura para F3 como uma variável Global;*

**Variáveis Locais – F3\_CAMonFly – Child Programa SFC**

*boCAM\_Axis1OK : BOOL ; //* *Indica que a tabela de perfis CAM para Eixo 1 foi calculada;*

*boCAM\_Axis2OK : BOOL ; //* *Indica que a tabela de perfis CAM para Eixo 2 foi calculada;*

*boChangeVel : BOOL ; //Flag auxiliar para indicar mudanças de velodidade no VM;*

*Inst\_MC\_SetPos\_Axis1 : MC\_SetPos ; //* *Ponto de ajuste do tempo para o movimento Eixo1 de P1-P2;*

*Inst\_MC\_SetPos\_Axis2 : MC\_SetPos ; //* *Define a posição inicial (P1) para o eixo 2;*

*lrVM\_Velocity1 : LREAL ; //* *Velocidade de comando para o Mestre Virtual;*

*MC\_CamIn\_Axis1 : MC\_CamIn ; //* *O Eixo 1 segue a VM com base na tabela Cam;*

*MC\_CamIn\_Axis2 : MC\_CamIn ; //* *O Eixo 2 segue a VM com base na tabela Cam;*

*MC\_Halt\_VM : MC\_Halt ; //* *Bloco de funções para desacelerar a VM para zero;*

*MC\_Halt\_Axis1\_1 : MC\_Halt ; //* *Bloco de função para desacelerar o Eixo1 para zero;*

*MC\_Halt\_Axis2\_1 : MC\_Halt ; // Bloco de função para desacelerar o Eixo1 para zero;*

*MC\_MoveVelocity\_VM : MC\_MoveVelocity ; //Velocidade do Mestre Virtual;*

*MC\_Power\_VM : MC\_Power ; //* *MC\_Power para ativar o mestre virtual;*

*MCFB\_CAMSwitch\_Axis1 : MCFB\_CAMSwitch ; //* *Constrói uma curva CAM on-line no Eixo 1;*

*MCFB\_CAMSwitch\_Axis2 : MCFB\_CAMSwitch ; //* *Constrói uma curva CAM on-line no Eixo 2;*

*TON\_StartF3 : TON ; //* *Temporizador atrasa inicialização do F3;*

*VelTrigger : r\_trig ; //Flag que indica alterações na velocidade de comando da VM;*

**Estrutura de Variáveis – F3\_CAMonFly - Child Programa SFC**

*boModeON : BOOL ; //* *Variável usada para comandar/indicar que F3 está ON;*

*boModeOFF : BOOL ; // Variável usada para comandar/indicar que F3 está OFF;*

*boEngage\_Camming : BOOL ; //* *Habilita o came entre o Eixo 1 e o Eixo 2 para VM;*

*boCammingEngaged : BOOL ; //* *Indica que o Axis1 e o Axis2 estão acoplados no VM;*

*boStopEngagement : BOOL ; //* *Desacopla esse Axis1 e Axis2 da VM;*

*lrVM\_Velocity : LREAL := LREAL#10 ; //* *Velocidade comandada para VM;*

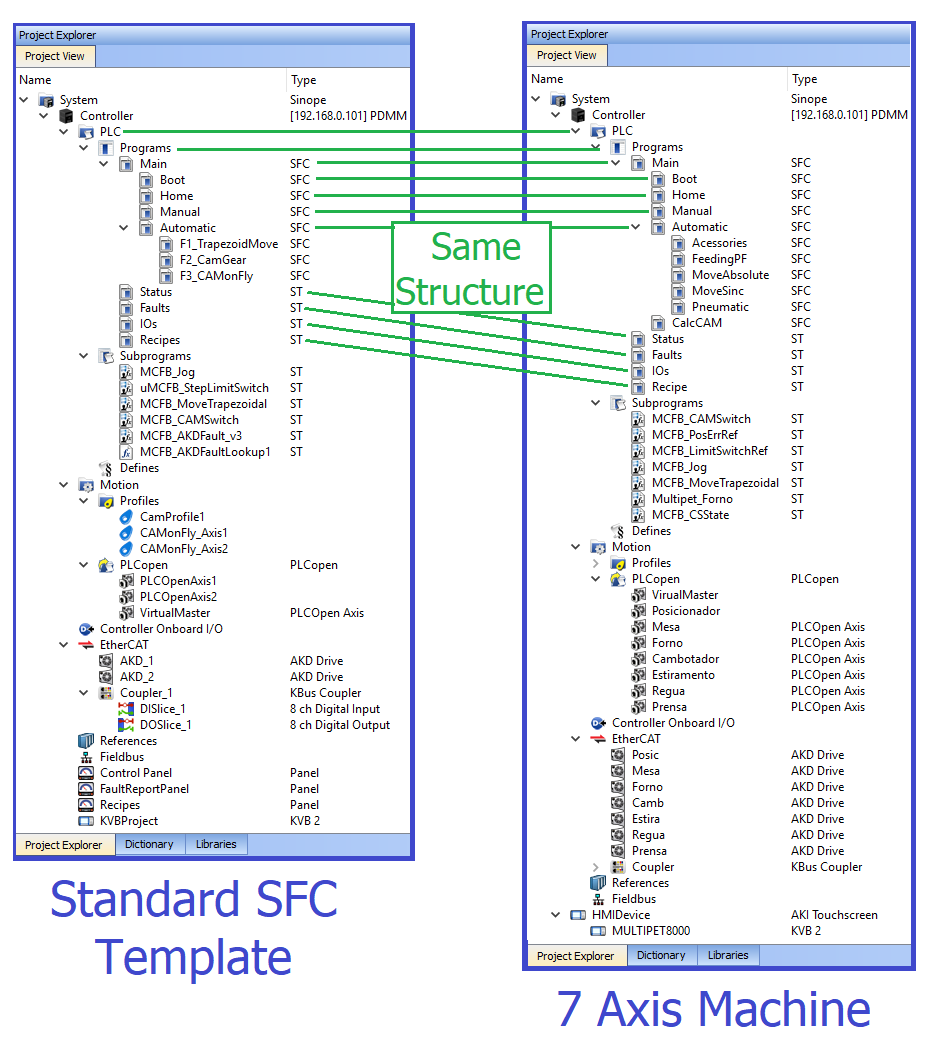
# Como escalonar o Template SFC para uma máquina de 7 eixos?

O Template foi projetado para permitir escalabilidade para projetos mais complexos. Fazendo uso da estrutura, o programador possui uma plataforma poderosa para organizar e estruturar o raciocínio, durante o desenvolvimento do projeto.

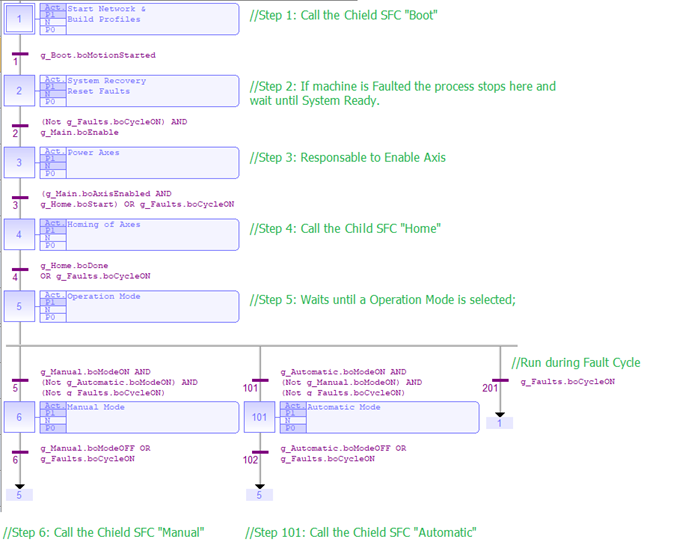
A divisão da máquina em funcionalidades e a organização do desenvolvimento em uma máquina de estado permite que todo o projeto seja estruturado antes do desenvolvimento da lógica. A estrutura do SFC ajuda o programador a identificar todas as etapas necessárias para iniciar e finalizar o projeto. Ele constrói todas as partes do projeto usando um ambiente gráfico e intuitivo.

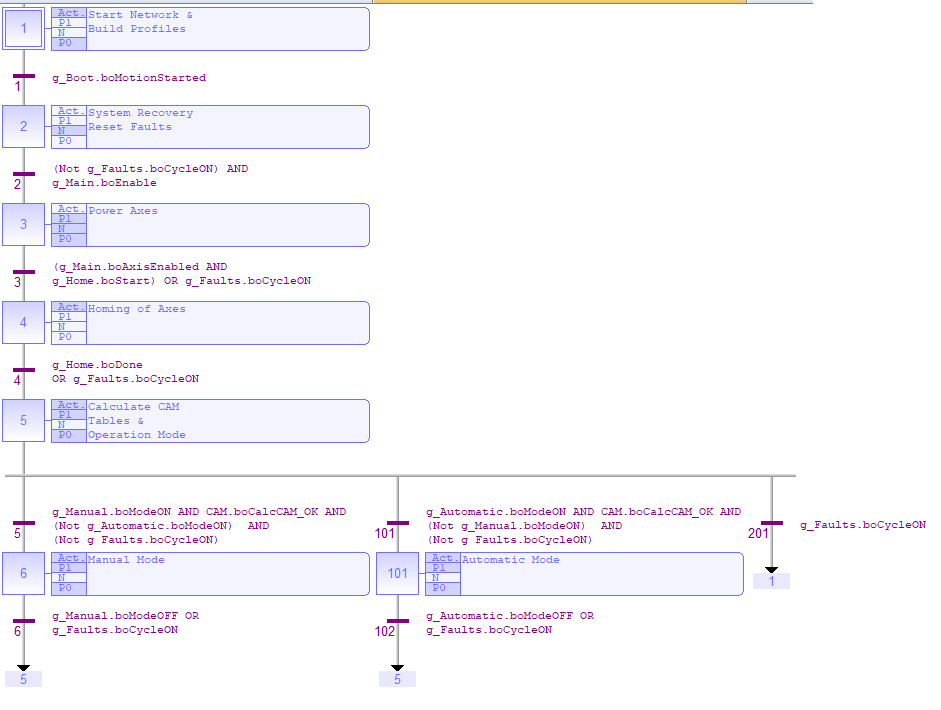
Nos tópicos a seguir, vemos o Template SFC padrão escalado para uma máquina de 7 eixos.

## Repartição do Projeto

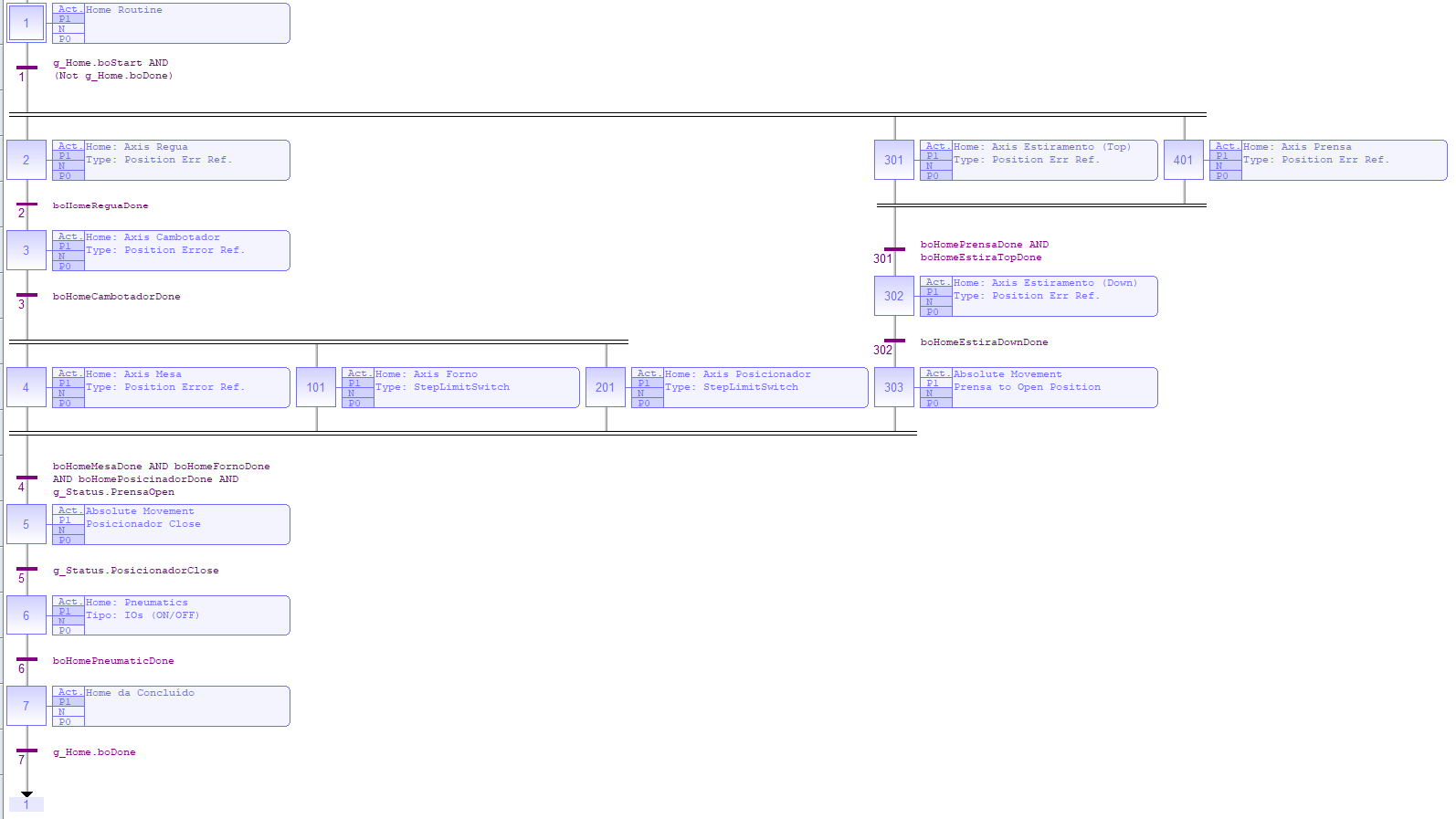


## Rotina Main



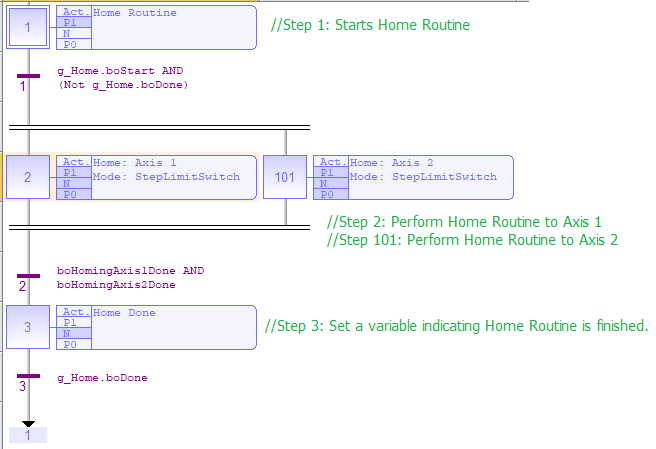


## Rotina de Homing









## Modo Automático

