WEINTEK LABS., INC.

Recipe Database

Backup Recipe Database to USB Drive

Demo Project

Release Date

2014/10/14

Contents

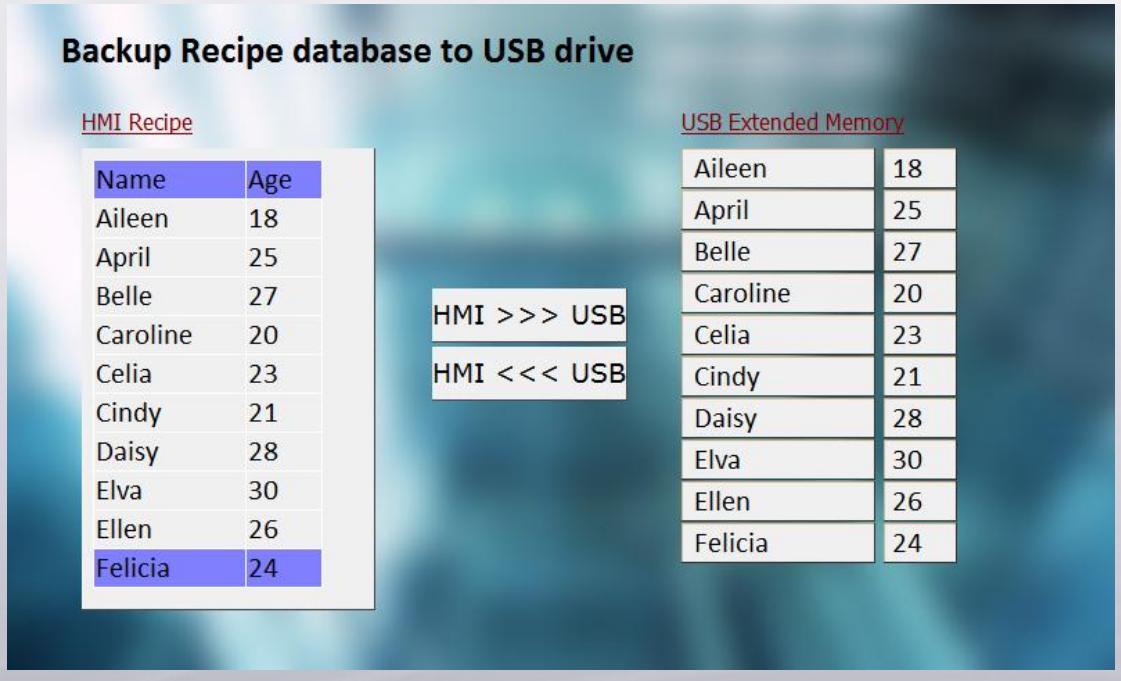
1. Overview and Operation

Overview

The Recipe Database backup (.db file) saved in the USB drive cannot be used to update the database on the original HMI, or transferred between HMIs. This demo project introduces how to convert the .db file into .emi file saved in the extended memory, i.e. USB drive, and transfer the data between HMIs.

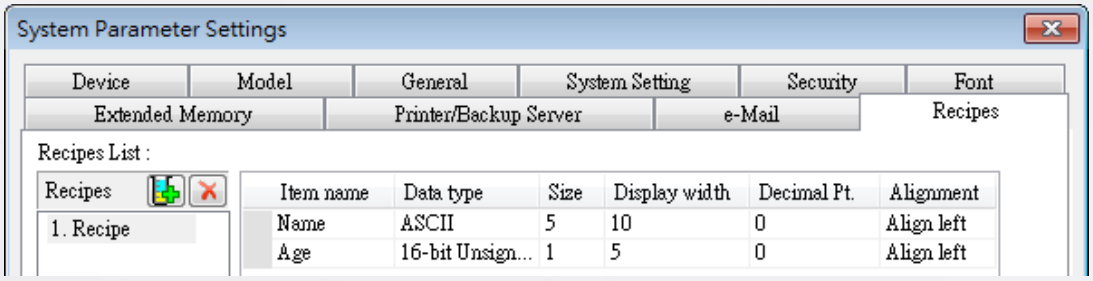
Operation

Launch EasyBuilder, click [Tools] » [Off-line Simulation] to run the demo project. The layout of the project is shown in the following figure, click [HMI>>>USB] to back up the recipe database on HMI to USB drive, or click [HMI<<<USB] to transfer the .emi file saved in the USB drive to the recipe database on HMI.



2. Setting up the Screen

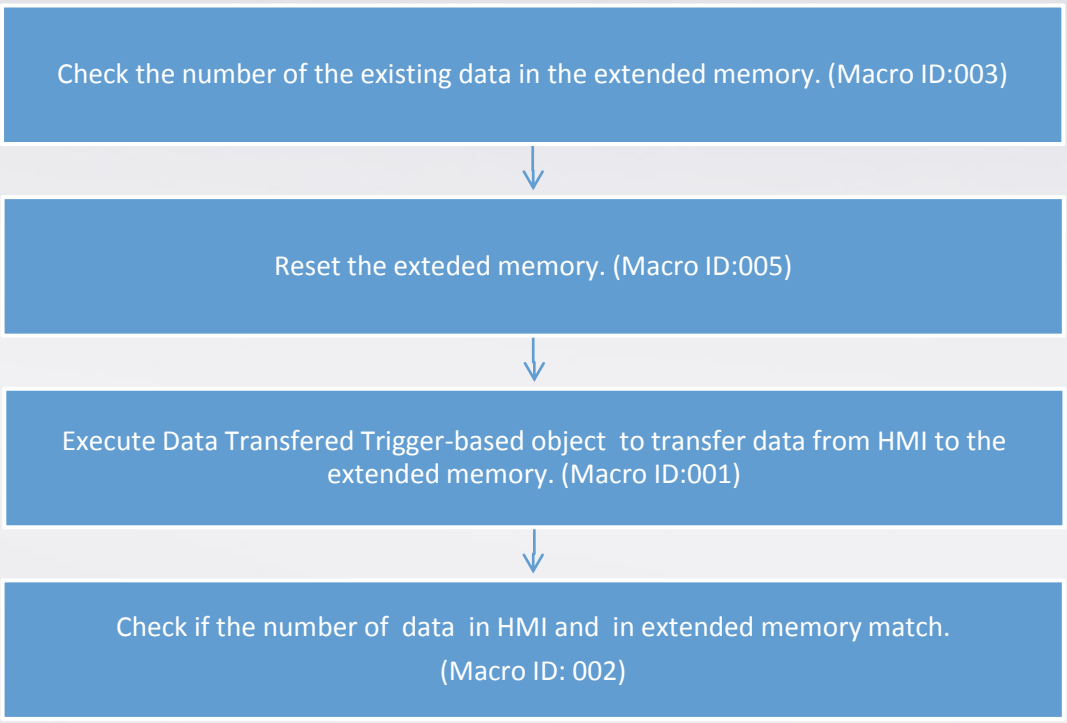
Step 1. Create recipe items in [System Parameter Settings] » [Recipes] tab and then add recipe contents in [Library] » [Recipe Records]. In this demo project, the items are set as shown in the following figure.



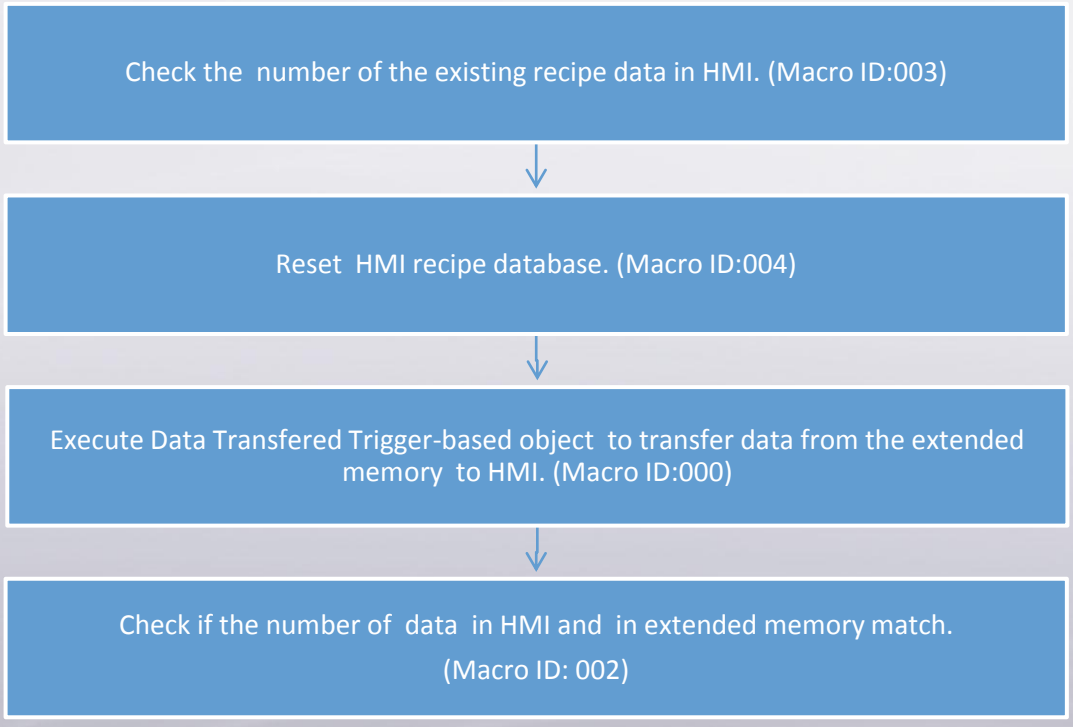
Step 2. Create two Data Transfer Trigger-based objects. One is for transferring the recipe database from HMI to USB driver, and the other goes the opposite way.

Step 3. Create ASCII Display objects and Numeric Display objects to display the data in the extended memory. Please Note that the layout of the objects must be identical to the recipe database on HMI. As shown in this demo project, each row has one ASCII object, set to ASCII data type, length 5, and one Numeric object, set to 16-bit Unsigned.

Step 4. The recipe data on HMI is transferred to USB drive in the following procedure.



Step 5. The recipe data in USB drive is transferred to HMI in the following procedure.



3. Addresses

The addresses of objects used in this demonstration are listed below.

Object	Address	Object ID	Description
Window 10			
Recipe View		RV_0	Displays Recipe Database.
Data Transfer Trigger-based	LB-0	RP_0	Transfers recipe data from HMI to USB drive.
Data Transfer Trigger-based	LB-1	RP_1	Transfers recipe data from USB drive to HMI.
Function Key		FK_1	Triggers Macro ID:001
Function Key		FK_0	Triggers Macro ID:000
ASCII Display	EM-0	AE_0	Displays recipe data in USB drive. (5 Words)
Numeric Display	EM-5	NE_0	Displays recipe data in USB drive.
ASCII Display	EM-6	AE_0	Displays recipe data in USB drive. (5 Words)
Numeric Display	EM-11	NE_0	Displays recipe data in USB drive.
Other similar objects are omitted in this list.			
Window 13			
Word Lamp	LW-0	WL_0	Transferring Status 0: Transferring Please wait… 1: Reset DB 2: Reset EM 3: Please wait… 4: Transferring from USB to HMI 5: Transferring from HMI to USB
Flow Block		FB_0	
Bar Graph	LW-12	BG_0	Displays percentage of completion.
Numeric Display	LW-10	ND_0	Displays percentage of completion.
Window 14			
Word Lamp	LW-0	WL_0	Displays transferring status.

Set Bit	LB-14	SB_0	Closes Direct Window.
Other Addresses			
	LW-2		Used for Macro. Records the number of recipe data in the extended memory, 6words as one unit (5 words + one 16-bit unsigned, same as the HMI recipe database).
	LW-4		Used for Macro. Records the number of recipe data in the extended memory.
	LW-9201		Index register 1. The register controls Data Transfer Trigger-based object.

4. Macro

1. Macro ID 000: Transfers recipe data from USB to HMI.

```
short db_count //Recipe.Count
short db_selection = 0 //Recipe.selection
short add = 1, update = 2, delete = 3 //Recipe.comamand
short i
bool on = true, off = false
short EMNo
short EMSize = 0
short data[12]
short status_message //0: Transferring 1: Reset DB  2: Reset EM  3.
short reset = 0

//Initial
SetData(on, "Local HMI", "Trasnferring Status", 1)
SetData(off, "Local HMI", "Error", 1)
status_message = 3
SetData(status_message, "Local HMI", "Status Message", 1)
SetData(reset, "Local HMI", "Progress", 1)
SetData(db_count, "Local HMI", "Progress_BarGraph", 1)

////EM Info

SYNC_TRIG_MACRO(3)

////Reset Recipe database
status_message = 1
SetData(status_message, "Local HMI", "Status Message", 1)
SYNC_TRIG_MACRO(4)

//Start Uploading
status_message = 0
SetData(status_message, "Local HMI", "Status Message", 1)
GetData(EMNo, "Local HMI", "EMNo", 1)

for i = 0 to emNo - 1

    //Update Progress
    UpdateProgress(i, emNo)
    //Source Data
    EMSize = i * 6
    SetData(EMSize, "Local HMI", LW, 9201, 1)
    //Destination Data
```

```
    DELAY(50)
    SetData(on, "Local HMI", "Data Transfer: USB > HMI", 1)
    DELAY(500)
    SetData(off, "Local HMI", "Data Transfer: USB > HMI", 1)
    SetData(add, "Local HMI", RECIPE, "Recipe.Command")

    DELAY(1000)

next

//Transferring completed, close popup window
SetData(off, "Local HMI", "Trasnferring Status", 1)

//Amount Check
SYNC_TRIG_MACRO(2)
```

2. Macro ID 001: Transfers recipe data from HMI to USB.

```
short db_count //Recipe.Count
short db_selection = 0 //Recipe.selection
short add = 1, update = 2, delete = 3 //Recipe.comamand
short i
bool on = true, off = false
short EMNo
short EMSize = 0
short data[12]
short status_message //0: Transferring 1: Reset DB  2: Reset EM  3.
short reset = 0

//Initial
SetData(on, "Local HMI", "Trasnferring Status", 1)

SetData(off, "Local HMI", "Error", 1)
status_message = 3
SetData(status_message, "Local HMI", "Status Message", 1)
SetData(reset, "Local HMI", "Progress", 1)
SetData(db_count, "Local HMI", "Progress_BarGraph", 1)

////Reset EM
status_message = 3
SetData(status_message, "Local HMI", "Status Message", 1)
SYNC_TRIG_MACRO(3)

////Reset EM
status_message = 2
SetData(status_message, "Local HMI", "Status Message", 1)
SYNC_TRIG_MACRO(5)

//Start Uploading
status_message = 0
SetData(status_message, "Local HMI", "Status Message", 1)
GetData(db_count, "Local HMI", RECIPE, "Recipe.Count")
```

```
for i = 0 to db_count - 1

    //Update Progress
    UpdateProgress(i, db_count)
    //Source Data
    SetData(i, "Local HMI", RECIPE, "Recipe.Selection")
    //Destination Data
    EMSize = i * 6
    SetData(EMSize, "Local HMI", LW, 9201, 1)
    //Transfer
    DELAY(500)
    SetData(on, "Local HMI", "Data Transfer: HMI > USB", 1)
    DELAY(500)
    SetData(off, "Local HMI", "Data Transfer: HMI > USB", 1)

next

//Transferring completed, close popup window
SetData(off, "Local HMI", "Trasnferring Status", 1)

//Transferring completed, close popup window
SetData(off, "Local HMI", "Trasnferring Status", 1)

//Amount Check
SYNC_TRIG_MACRO(2)
```

3. Macro ID 002: Compares the number of recipe data in HMI recipe database and in extended memory.

```
short EMNo
short db_count
short status_message
bool on = true, off = false

SYNC_TRIG_MACRO(3)
GetData(db_count, "Local HMI", RECIPE, "Recipe.Count")
GetData(EMNo, "Local HMI", "EMNo", 1)

    if EMNo > db_count then
    SetData(on, "Local HMI", "Error", 1)
    status_message = 4
    SetData(status_message, "Local HMI", "Status Message", 1)
    end if

    if EMNo < db_count then
    SetData(on, "Local HMI", "Error", 1)
    status_message = 5
    SetData(status_message, "Local HMI", "Status Message", 1)
    end if
```

4. Macro ID 003: The number of recipe data in the extended memory.

```
short EMNo, EMSize, data

for EMSize = 0 to 600 step 6

    GetData(data, "Local HMI", EM0, EMSize, 1)

    DELAY(50)
    SetData(EMSize, "Local HMI", "EMSize", 1)
    EMNo = EMSize / 6
    SetData(EMNo, "Local HMI", LW, 200, 1)

    if data == 0 then
    break
    end if

next

    SetData(EMSize, "Local HMI", "EMSize", 1)
    SetData(EMNo, "Local HMI", "EMNo", 1)
```

5. Macro ID 004: Resets recipe database.

```
short db_count //Recipe.Count
short db_selection = 0 //Recipe.selection
short delete = 3 //Recipe.comamand
short recordID = 0

GetData(db_count, "Local HMI", RECIPE, "Recipe.Count")

for db_selection = 0 to db_count - 1

    SetData(recordID, "Local HMI", RECIPE, "Recipe.Selection")
    SetData(delete, "Local HMI", RECIPE, "Recipe.Command")
    DELAY(500)

    UpdateProgress(db_selection, db_count)

next
```

6. Macro ID 005: Resets extended memory.

```
short EMNo, EMSize
short data[6]
short i, j

GetData(EMNo, "Local HMI", "EMNo", 1)
GetData(EMSize, "Local HMI", "EMSize", 1)

FILL(data[0], 0, 6)

for i = 0 to EMSize step 6

SetData(data[0], "Local HMI", EM0, i, 6)
DELAY(50)

j = i/6
UpdateProgress(j, EMNo)

next
```

7. Macro sub function: Calculates the percentage of completion.

```
sub UpdateProgress(short SerialNumber, short TotalNumber)

//SerialNumber: The serial number of current data transmitted.
//TotalNumber: The total amount of data.
//Progress: The amount of data transmitted. (Serial Number / Total Number)

short Progress, Progress_BarGraph = 100

Progress = SerialNumber * 100 / TotalNumber
Progress_BarGraph = 100 - Progress

SetData(Progress, "Local HMI", "Progress", 1)
SetData(Progress_BarGraph, "Local HMI", "Progress_BarGraph", 1)

end sub
```