AfficheBin

Déclarer i comme uint8_t avec valeur 0

Déclarer nbBits comme uint8_t avec valeur 0

Déclarer cntBin comme uint16_t

Déclarer cntTrigo comme uint16_t

Déclarer values comme stValues

Pour i plus petit que 32

- values.tbBin[i] = 0

Si mode > 3

V

- mode = 0

F

- ∅

Si mode différent de 0

V

- userVal = (int)userVal

F

- ∅

Appeler la fonction ConvBin avec les paramètres userVal, tbBin, sizeTb

switch case mode

Cas 0	Cas 1	Cas 2	Cas 3	
nbBits = 32	nbBits = 8	nbBits = 16	nbBits = 32	break
Tant que tbBin[nbBits - 1] == 0				
nbBits décrémente				

values.nbBits = nbBits

Pour i allant de nbBits à 1

Si tbBin[i - 1] == 46 (code ASCII du point)

V

- Afficher "."
- values.tbBin[i - 1] = '.'

F

- Afficher la valeur du bit (tbBin[i - 1])
- values.tbBin[i - 1] = tbBin[i - 1]

values.userVal = userVal

values.mode = 0

Appeler lectureCntIterations avec les arguments cntBin et cntTrigo

cntBin s'incrémente

Appeler ecritureFichierLogs avec les arguments cntBin, cntTrigo et values