SWR POWER METER F8KGL

Description et spécifications
V 0.5

F0EOS-F4BJH-21/12/2017-Vauréal Amitié Radio

1-INTRODUCTION

1.1-Généralités

Le « SWR Power Meter F8KGL » est un dispositif de mesure dédié aux applications radioamateur, permettant de mesurer, en émission, la puissance transmise, la puissance réfléchie et le ROS.

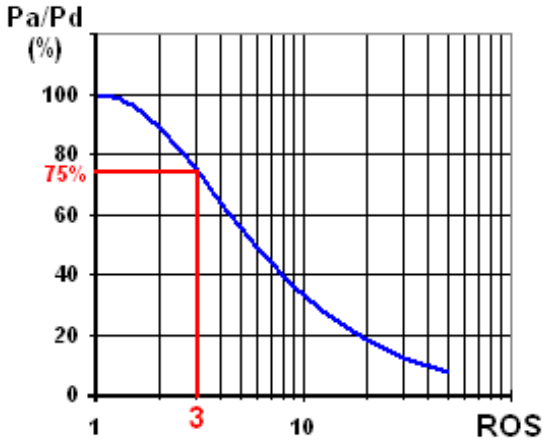
Afin d'optimiser le transfert de puissance d'un émetteur vers l'antenne, l'OM doit rechercher le maximum de puissance transmise pour le minimum de puissance réfléchie. Ce point de fonctionnement est atteint lorsque les impédances de la source et de la charge sont strictement identiques. On dit alors que les impédances sont adaptées.

Le « SWR Power Meter F8KGL » est donc un dispositif permettant, d'une part, de mesurer la puissance transmise et la puissance réfléchie. Il mesure également le « SWR » (ou ROS), pour « Standing Wave Ratio » (ou Rapport d'Onde Stationnaire).

$$SWR = \frac{Ptranmise + Préfléchie}{Ptransmise - Préfléchie}$$

La courbe ci-dessous (source F5ZV) montre que la baisse de puissance est insignifiante tant que le ROS ne dépasse pas 2 et reste acceptable pour un ROS de 3. La baisse de signal en dB par rapport à la pleine puissance est à peine visible chez le correspondant :
- ROS=1,5 : puissance non absorbée=4% (0,2dB)
- ROS=2 : puissance non absorbée=11% (0,5dB)
- ROS=3 : puissance non absorbée=25% (1,2dB)

Sur décamétriques, ou lorsque la liaison est confortable, perdre 1dB n'est pas un souci. Par contre, lorsque l'on chasse le décibel comme en trafic EME (par réflexion sur la Lune), un ROS de 2 est déjà inacceptable.



Cette perte de puissance, que l'on doit distinguer de la perte d'énergie due au courant plus élevé correspondant à l'onde réfléchie, peut être exprimée en décibels.
Exemple : avec un ROS de 3, un émetteur qui pourrait débiter 100 watts (dans une antenne bien adaptée) ne fournira que 75 watts.

1.2-Ligne de mesure

Pour mesurer la puissance d'un signal radioélectrique, plusieurs techniques sont possibles, mais la plus utilisées reste la technique de « la diode de détection HF ».
De plus, pour discrimer l'onde incidente de l'onde réfléchie, nous devons utiliser un « coupleur directionnel »

1.3-Fonctionnement

A la mise sous tension, le « SWR-POWER METER F8KGL », affiche le message suivant pendant 5s :

S	W	R	-	P	O	W	E	R		m	e	t	e	r	
			F	8	K	G	L		V	n	.	m			

Vn.m correspond à la version du logiciel chargée dans la mémoire du microcontrôleur.

Au boot des 5 secondes, le « SWR-POWER METER F8KGL » détermine s'il est en mode « calibration », ou en mode « mesure »

En mode calibration, le « SWR-POWER METER F8KGL » affiche :

A	D	C	f	w	d		x	x	x	x					
A	D	C	r	e	f		y	y	y	y					

En mode « mesure », le « SWR-POWER METER F8KGL » affiche :

F	W	D			R	E	F			S	W	R		
a	a	a	W		b	b	b	W		c	.	c	c	!	!

« aaa » correspond à la mesure de la puissance transmise en W
« bbb » correspond à la mesure de la puissance réfléchie en W
« c.cc » correspond à la mesure du ROS. 2 points d’exclamation clignotant s’affichent « !! » si le ROS>2

2-MATERIEL

2.1-Description

Le coeur du dispositif repose sur l'emploi d'un microcontrôleur PIC (fondeur Microchip) 16F88. Son choix a été guidé par ses principales caractérstiques suivantes :

PIC (+sa programmation) 4MHz
LCD
http://digitaldiy.io/articles/mcu-programming/assembly/55-assembly-example/114-mpasm-tutorial-liquid-crystal-display-lcd#.Wi5383mDO9I
Ligne de mesure
ADC
Boitier
Alimentation

2.1.1-LCD

2.1.2Ligne de mesure

HSMS tartempion

2.1.3-ADC

Résolution 10 btis

2.1.3-Alimentation

Pile + connnecteyr

2.1.3-Boitier

Dimensions ?
Matière
crosbar

2.2-Schéma fonctionnel

2.x-Calibration

Le prototype servira d'étalon pour l'ensemble des dispositifs.

Sur le 1er prototype :

1. Relevé de la courbe P=f(ADC) avec une application de calibration, et « le petit détecteur à diode home-made »
   Schéma du petit détecteur home made :

   Schéma du dispositif de mesure

2. Estimer de la manière la plus précise possible, les pertes de la ligne de couplage (estimé à environ 60dB?)

3. Trouver 3 points de calibration (1, 5W, 50W), qui permettent d'interpoler par morceaux linéraires la courbe P=f(ADC)

Les 3 courbes ainsi définies serviront de base à la calibration de tous les autres dispositifs. Noter les 3 équations ainsi obtenues.

Pour la la calibration des autres dispositifs

- Placer le dispositif en mode calibration
- Relever les valeurs de l'ADC aux points 1, 5W, 50W (identiques à ceux trouvés précédemments)
- Ajuster les 3 équations précentes le cas échéant
- Dresser la table de calibration
  0x0001 = 1mW
  0x0002 = 1,1mW
  etc.
- Table à charger en EEPROM

3-SPECIFICATIONS

3.1-Généralités

L'architecture hardware étant fondée sur le PIC 16F88 de chez Microchip, le langage de développement du logiciel sera l'assembleur.
Le logiciel sera développé sous Linux (Ubuntu 16.04 ou Debian 8).

3.1.1-Application

L'application sera généré au format « .hex » pour être programmée dans la mémoire du microcontrôleur : « swr_power_meter_f8kgl-Vn.m.hex »

Elle contiendra, essentiellement, 2 fonctionnalités :

- Fonctionnalité de calibration

A l'aide 3 points de mesures (1W, 5, 50W), interpolation de la courbe de calibration
Définir la procédure de calibration :
-avec le « petit détecteur maison », faire le relevé de la courbe P = f(ADC) sur les 2 ports FWD et REF
-avec la ligne de couplage complète, relever 3 points de mesures ADC = f(P)
-faire une interpolation entre les points qui tend à s'approcher de la 1ère courbe

TBD

- Fonctionnalité de mesure de la puissance transmise, de la puissance réfléchie, et le calcul du SWR

TBD

3.2-Outils de développement

3.2.1-Assembleur

Sous linux, la suite « GPUTILS », permet la compilation d'un projet développé en assembleur pour PIC.

GPUTILS est une collection d'outil pour les microcontroleurs PIC. Elle inclut :

- Gpasm : compilateur assembleur
- Gplib : compilateur assembleur permettant la génération d'une librairie
- Gplink : éditeur de lien symboliquez

Installation :

1. Désinstaller la version courante de la distribution
2. Télécharger la version 1.5.0-1 en suivant ce lien : https://sourceforge.net/projects/gputils/files/gputils/1.5.0/gputils-1.5.0-1.tar.gz/download

3. Installation

```
$ tar –xvzf gputils-1.5.0-1.tar.gz
$ cd gputils-1.5.0-1.tar.gz
$ ./configure
$make
$sudo make install
```

3.2.2-Outils de validation

Sous linux, la suite « gpsim » permet la simulation d'un code compilé par GPUTILS

Installation :

1. Désinstaller la version courante de la distribution
2. Télécharger la version 0.30.0 en suivant ce lien : https://sourceforge.net/projects/gpsim/files/gpsim/0.30.0/gpsim-0.30.0.tar.gz/download
3. Installation

```
$ tar –xvzf gpsim-0.30.0.tar.gz
$ cd gpsim-0.30.0.tar.gz
$ ./configure
$make
$sudo make install
```

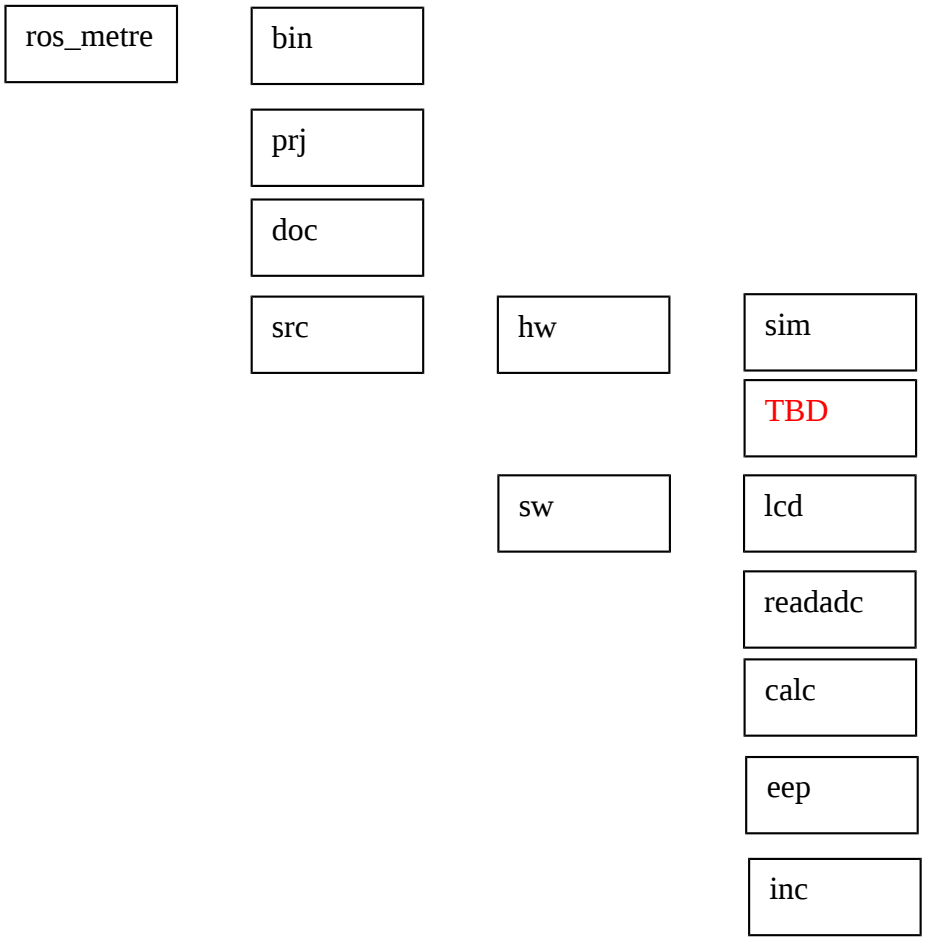
Utilisation :

1.
```
$ gpsim –s nom_du_fichier.cod
```

2. Aller dans File->Open et choisir le fichier .stc
3. Par défaut, la fréquence est fixée à 20MHz. Il faut fixer la fréquence de travail à 4MHz :

```
**gpsim> frequency 4000000
**gpsim> frequency
Clock frequency: 4 MHz.
```

3.3-Architecture SW



swr_power_meter/bin	Contient l'ensemble des binaires produits : • *.a : librairie associée à un composant sw • *.cod : simulation • *.hex : binaire à flasher dans le PIC • *.map : mapping mémoire • *.cof : fichier objet résultat de la compilation • *.lst : ?
swr_power_meter/prj	Contient le Makefile du projet, et le point d'entrée sw (main.asm)
swr_power_meter/doc	Documentation du projet
swr_power_meter/src	Contient les sources du projet
swr_power_meter/src/hw	Contient les sources HW du projet
swr_power_meter/hw/sim	Contient le fichier netlist pour gpsim
swr_power_meter/hw/TBD	Contient les schéma, routages, gerber etc
swr_power_meter/sw/lcd	Composant LCD • Driver.asm : driver bas niveau du LCD • Aff.asm : routines haut niveau d'affichage des messages • Makefile : make de la librairie LCD
swr_power_meter/sw/readadc	Composant ADC
swr_power_meter/sw/calc	Composant CALC
swr_power_meter/sw/eep	Composant EEP : fonctionnalité EEPROM
Ros_metre/sw/inc	Include

3.4-Architecture HW

3.5-Spécifications

3.5.1-/prj

3.5.1.1-Makefile

3.5.1.2-Main.asm

Fonctions	Fonction principale, point d'entrée du logiciel
Nom	Init
Version	≥0.1 (sauf les TBD)
Paramètres entrée	
Paramètres sorties	

Traitements	• Initialisation
	  ◦ PIC : effectuer l'initialisation du PIC
	  ◦ LCD : Effectuer l'initialisation du LCD (lcd_init)
	  ◦ ADC : Effecteur l'initialisation de l'ADC
	  ◦
	• Afficher le message de boot (lcd_affbootmsg)
	• Tempo de 5s
	  ◦ temporisation de 2,5s (tempo_boot)
	  ◦ temporisation de 2,5s (tempo_boot)
	• Effacer le LCD (lcd_clear)
	• Positionner le curseur du LCD sur la ligne 1
	• Tester le mode calibration (TBD teste octet en EEPROM)
	• Si le boîtier est en mode de calibration (TBD)
	  ◦ afficher le message de calibration (lcd_affcalib)
	  ◦ Dans une boucle infinie
	  ▪ lire les registres ADCfwd et ADCref (adc_readAN0, adc_readAN1)
	  ▪ afficher le message de mesure (lcd_affadc)
	• Sinon
	  ◦ Dans une boucle infinie (TBD) :
	  ▪ lire les registres ADCfwd et ADCref
	  ▪ calculer la puissance FWD et REF
	  ▪ Calculer le SWR
	  ▪ Afficher le message de mesure

Fonctions	Temporisation de 2,5 secondes
Nom	tempo_boot
Version	≥0.2
Paramètres entrée	

Paramètres sorties	
Traitements	Appeler 10 fois un délai de 250ms

3.5.2/sw/inc

3.5.2.1-lcd.inc

Contient les define du LCD.

3.5.2.2-eep.inc

Contient le plan mémoire de l'EEPROM

__EEPROM_START	Version du logiciel
__EEPROM_START + 5	Mode calibration (0x00) ou mode mesure (0xFF)
__EEPROM_START + 6	Table de calibration

3.5.3-/sw/lcd

3.5.3.1-driver.asm :

Fonction	Initialisation du LCD
Version	≥0.1
Nom	lcd_init
Paramètres entrée	
Paramètres sorties	
Traitements	

Fonction	Affichage d'un caractère
Version	≥0.1
Nom	lcd_affchar
Paramètres entrée	W(1 byte) : contient le caractère à afficher à la position courante du curseur
Paramètres sorties	
Traitements	

Fonction	Envoi d'une commande au LCD
Version	≥0.1
Nom	lcd_sendcmd
Paramètres entrée	W(1 byte) : contient la commande 0x28 Set Interface Length

	0x10 Turn Off Display 0x01 Clear Display RAM 0x06 Set Cursor Movement 0x0C Turn on Display/Cursor 0x01 Clear display 0xc0 move to 2nd row, first column
Paramètres sorties	
Traitements	

Fonction	Positionner le curseur du LCD
Nom	lcd_setposcursor
Version	≥0.1
Paramètres entrée	W(1 byte) : contient la position du curseur 0-15 : 1ère ligne 16-31 : 2ème ligne
Paramètres sorties	
Traitements	1. Si le curseur doit être positionné sur la première ligne : W = W + 0x80 Si le curseur doit être positionné sur la deuxième ligne : W = W + 0xC0 2. Envoi de la commande au LCD (lcd_sendcmd)

Fonction	Efface le LCD
Version	≥0.1
Nom	lcd_clear
Paramètres entrée	
Paramètres sorties	
Traitement	1. W=0x01 2. Envoi de la commande au LCD (lcd_sendcmd)

Fonction	Positionne le curseur sur la 2ème ligne
Version	≥0.1
Nom	lcd_setposL2
Paramètres entrée	
Paramètres sorties	
Traitements	1. W=0xC0 2. Envoi de la commande au LCD (lcd_sendcmd)

Fonction	Conversion hexa-ASCII
Version	≥0.1

Nom	lcd_convtoascii
Paramètres entrée	W (1 quartet) : contient le quartet de poids faible à convertir
Paramètres sorties	W (1 byte) : contient l'octet converti
Traitements	W=W + 0x30

Fonction	Routines de temporisation et pulse
Version	≥0.1
Nom	
Paramètres entrée	
Paramètres sorties	
Traitements	http://digitaldiy.io/articles/mcu-programming/assembly/55-assembly-example/114-mpasm-tutorial-liquid-crystal-display-lcd#.Wi5383mDO9I

3.5.3.2-aff.asm :

Fonction	Message de version courante du logiciel
Version	≥0.1
Nom	c_swversion
Paramètres entrée	
Paramètres sorties	
Traitements	Zone mémoire (5 bytes) dédiée au stockage de la version du logiciel « Vn.m »,0x00 Cette zone de mémoire est placée au début de l'EEPROM (0x2100). Cette zone mémoire doit se terminer par l'octet 0x00. Cette zone mémoire est remplie par le compilateur.

Fonctions	Message de boot ligne 1 du LCD
Version	≥0.1
Nom	bootmsgL1
Paramètres entrée	v_charpos : position du caractère à retourner
Paramètres sorties	W (1 byte) : contient le caractère ou 0x00 si pas de caractère
Traitements	Zone mémoire dédiée au stockage du message de boot (ligne 1 du LCD) contenant la chaîne suivante : « SWR-POWER meter » • Additionner le pointeur de programme avec v_charpos • Retourner le caractère contenu en mémoire à cette position dans W • Fin de chaîne = retourner 0x00 dans W

Fonctions	Message de boot ligne 2 du LCD
Version	≥0.1
Nom	bootmsgL2
Paramètres entrée	v_charpos : position du caractère à retourner
Paramètres sorties	W (1 byte) : contient le caractère ou 0x00 si pas de caractère
Traitements	Zone mémoire dédiée au stockage du message de boot (ligne 2 du LCD) contenant la chaîne suivante : « F8KGL » • Additionner le pointeur de programme avec v_charpos • Retourner le caractère contenu en mémoire à cette position dans W • Fin de chaîne = retourner 0x00 dans W
Fonctions	Message de calibration L1 du LCD
Version	≥0.2
Nom	calibmsgL1
Paramètres entrée	v_charpos : position du caractère à retourner
Paramètres sorties	W (1 byte) : contient le caractère ou 0x00 si pas de caractère
Traitements	Zone mémoire dédiée au stockage du message de calibration (ligne 1 du LCD) contenant la chaîne suivante : « ADCfwd » *avec un espace à la fin* • Additionner le pointeur de programme avec v_charpos • Retourner le caractère contenu en mémoire à cette position dans W • Fin de chaîne = retourner 0x00 dans W

Fonctions	Message de calibration ligne 2 du LCD
Version	≥0.2
Nom	calibmsgL2
Paramètres entrée	v_charpos : position du caractère à retourner
Paramètres sorties	W (1 byte) : contient le caractère ou 0x00 si pas de caractère
Traitements	Zone mémoire dédiée au stockage du message de boot (ligne 2 du LCD) contenant la chaîne suivante : « ADCref » *avec un espace à la fin* • Additionner le pointeur de programme avec v_charpos • Retourner le caractère contenu en mémoire à cette position dans W • Fin de chaîne = retourner 0x00 dans W

Fonction	Affichage du message de boot
Version	≥0.1
Nom	lcd_affboot
Paramètres entrée	-bootmsgL1 : zone mémoire (15 bytes) contenant le message de boot ligne 1 -bootmsgL2 : zone mémoire (5 bytes) contenant le message de boot ligne 2 -c_swversion : zone EEPROM (5bytes) contenant la version courante du logicielle
Paramètres sorties	S W R - P O W E R m e t e r F 8 K G L V 0 . 5
Traitements	1. v_charpos = 0x00 2. Afficher le message de boot ligne 1 Tant que W≠0 o Récupérer 1 caractère du message de boot ligne 1 (bootmsgL1) dans W o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos 3. Positionner le curseur sur la ligne 2, 4ème case : o W=0x13 o Positionner le curseur du LCD (lcd_setposcursor) 4. v_charpos = 0x00 5. Afficher le message de boot ligne 2 Tant que W≠0 o Récupérer 1 caractère du message de boot ligne 2 (bootmsgL2) o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos 6. Positionner le curseur sur la ligne 2, 10ème case : o W=0x1C o Positionner le curseur du LCD (lcd_setposcursor) 7. v_charpos = 0x00 8. afficher la version Tant que W≠0 o Récupérer le caractère en EEPROM W ← v_charpos EEADDR ← W Lire un octet en EEPROM (eep_readbyte) o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos 9. FIN

Fonction	Affichage du message de calibration
Version	≥0.2
Nom	lcd_affcalib
Paramètres entrée	-calibADCL1 : zone mémoire (7bytes) contenant le message de calibration L1 -calibADCL2 : zone mémoire (7bytes) contenant le message de calibration L2

Paramètres sorties	A D C f w d A D C r e f
Traitements	1. v_charpos = 0x00 2. Afficher le message de calibration ligne 1 Tant que W≠0 o Récupérer 1 caractère du message de calibration ligne 1 (calibmsgL1) dans W o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos 3. Positionner le curseur sur la ligne 2 : o W=0x10 o Positionner le curseur du LCD (lcd_setposcursor) 4. v_charpos = 0x00 5. Afficher le message de calibration ligne 2 Tant que W≠0 o Récupérer 1 caractère du message de calibration ligne 2 (calibmsgL2) o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos

Fonction	Affichage d'1 octet en hexa sur le LCD
Version	≥0.4
Nom	lcd_affhexa
Paramètres entrée	W : contient l'octet en hexa à afficher
Paramètres sorties	
Traitements	1. v_tmp = W 2. swapper les quartes de v_tmp, et mettre le résultat dans W 3. Appliquer un masque sur les bits de poids faible 4. Convertir le quartet de poids faible en ASCII (lcd_convtoascii) 5. Afficher 1 caractère sur le LCD (lcd_affchar) 6. W=v_temp&0F 7. Convertir le quartet de poids faible en ASCII (lcd_convtoascii) 8. Afficher 1 caractère sur le LCD (lcd_affchar)

Fonction	Affichage de la mesure de calibration
Version	≥0.4
Nom	lcd_affadc
Paramètres entrée	v_adcfwd (2bytes) : résultat de l'ADC AN0 sur 10 bits v_adcref
Paramètres sorties	x x x x y y y y
Traitements	1.positionner le curseur sur la ligne 1, 8ème case

	2.W=v_adcfwd 3.Afficher un octet en hexa (lcd_affhexa) 4.W =v_adcfwd +1 5.Afficher un octet en hexa (lcd_affhexa) 6.positionner le curseur sur la ligne 2, 8ème case 7.W=v_adcref 8.Afficher un octet en hexa (lcd_affhexa) 9.W =v_adcref +1 10.Afficher un octet en hexa (lcd_affhexa)

Fonction	Affichage du message de calibration
Version	≥0.2
Nom	lcd_affcalib
Paramètres entrée	-calibADCL1 : zone mémoire (7bytes) contenant le message de calibration L1 -calibADCL2 : zone mémoire (7bytes) contenant le message de calibration L2
Paramètres sorties	A D C f w d A D C r e f
Traitements	6. v_charpos = 0x00 7. Afficher le message de calibration ligne 1 Tant que W≠0 o Récupérer 1 caractère du message de calibration ligne 1 (calibmsgL1) dans W o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos 8. Positionner le curseur sur la ligne 2 : o W=0x10 o Positionner le curseur du LCD (lcd_setposcursor) 9. v_charpos = 0x00 10. Afficher le message de calibration ligne 2 Tant que W≠0 o Récupérer 1 caractère du message de calibration ligne 2 (calibmsgL2) o Afficher 1 caractère sur le LCD (lcd_affchar) o Incrementer v_charpos

Fonction	Affichage de la puissance du port FWD TBD
Version	≥0.5
Nom	lcd_affpfwd
Paramètres entrée	v_pfwd
Paramètres sorties	
Traitements	

Fonction	Affichage de la puissance du port REF TBD
Version	≥0.5
Nom	lcd_affpref
Paramètres entrée	v_pref
Paramètres sorties	
Traitements	

Fonction	Affichage du SWR TBD
Version	≥0.5
Nom	lcd_affpref
Paramètres entrée	v_p_ref
Paramètres sorties	
Traitements	

3.5.4-/sw/calc

3.5.4.1-calc_swr

TBD

• Paramètres entrée	• P_FWD • P_REF
• Paramètres sorties	• SWR
• Traitements	• $SWR = \frac{ADCfwd + ADCref}{ADCfwd - ADCref}$

3.5.4.2-calc_power

TBD

Paramètres entrée	ADCfwd, ADCref table de calibration stockée en mémoire
Paramètres sorties	P_FWD P_REF
Traitements	Lire la valeurs dans les registres ADC récupérer la valeur correspondante dans la table de calibration

3.5.5-/sw/readadc

3.5.5.1-adc.asm

Fonction	Lire le résultat de la conversion A/N AN0
Version	≥0.4

Nom	adc_readAN0
Paramètres entrée	
Paramètres sorties	-v_adcfwd (2bytes) : résultat de l'ADC sur 10 bits
Traitements	1. Selectionner le canal à échantilloner (AN0) ADCON0(CHS2) = b'0' ADCON0(CHS1) = b'0' ADCON0(CHS0) =b'0' 2. Mise en service du convertisseur ADCON(ADON) = b'1' 3. Tempo de 20us 4.Lancer la phase de conversion ADCON0(G0)=b'1' 5. Tant ADCON0(G0)≠b'0' 6.v_adcfwd = ADRESH v_adcfwd(+1) = ADRESL

Fonction	Lire le résultat de la conversion A/N AN1
Version	≥0.4
Nom	adc_readAN1
Paramètres entrée	
Paramètres sorties	-v_adcref (2bytes) : résultat de l'ADC sur 10 bits
Traitements	1. Selectionner le canal à échantilloner (AN1) ADCON0(CHS2) = b'0' ADCON0(CHS1) = b'0' ADCON0(CHS0) =b'1' 2. Mise en service du convertisseur ADCON(ADON) = b'1' 3. Tempo de 20us 4.Lancer la phase de conversion ADCON0(G0)=b'1' 5. Tant ADCON0(G0)≠b'0' 6.v_adcref = ADRESH v_adcref(+1) = ADRESL

3.5.6-/sw/EEP/

3.5.6.1-eep.asm

Fonction	Lecture d'un octet en EEPROM
Version	≥0.3
Nom	eep_readbyte
Paramètres entrée	-W : contient l'adresse à lire en EEPROM
Paramètres sorties	-W : contient l'octet lu en EEPROM
Traitements	1. EADDR = W 2. EECON(RD) = b’1’ 2. W ← EEDATA

3.5.7-/src/hw/

3.5.7.1-schéma, brd

3.5.6.2-définition des ports du PIC (LCD, ADC, etc.)