Il pacchetto ecgdraw*

Marco Scavino† and Ezio Aimé‡

1 luglio 2016

Sommario

Lo scopo di questo pacchetto è la riproduzione di elettrocardiogrammi fittizi, appoggiandosi al pacchetto TikZ e al bundle LaTeX3.

Indice

1 Installazione

Per il corretto funzionamento del pacchetto, sono necessari:

- Il pacchetto TikZ,
- I pacchetti del bundle LaTeX3.

Per installare il pacchetto, è sufficiente compilare il file `ECG.dtx` e spostare i file `ECG.sty` e `archivio.tex` nella cartella che si interessa compilare o all'interno di cartelle raggiungibili dal sistema TeX.

*Versione numero v.1.0; ultima revisione 2016/06/29.

†e-mail: `scavino dot marco93 at gmail dot com`

‡e-mail: `ezio dot aime at fastwebnet dot it`

2 Uso

`ecg` Il pacchetto definisce l'ambinete `ecg` che accetta un comando facolativo, secondo la sintassi

`\begin{ecg}` `[`⟨*opzioni*⟩`]` tracciati ECG `\end{ecg}`

dove ⟨*opzioni*⟩ sono opzioni direttamente passate al pacchetto TikZ. All'interno dell'ambiente è possibile disegnare il diagramma tramite il comando `\ECG`, che si impiega tramite la sintassi

`\ECG` `[`⟨*opzioni*⟩`]` `(`⟨*posizione*⟩`)` `{`⟨*lista picchi*⟩`}`

Il comando accetta un comando opzionale `[`⟨*opzioni*⟩`]` dove è possibile elencare le opzioni che saranno passate a TikZ, un argomento tra parentesi tonde `(`⟨*posizione*⟩`)` anch'esso opzionale che permette di posizionare il picco e uno obbligatorio ⟨*lista picchi*⟩, dove sono presenti le sigle corrispondenti ai picchi da disegnare.

Ogni singola sigla deve essere scritta nell'ordine:

`\ECG { [`⟨*opzioni*⟩`]` `{`⟨*nome onda*⟩`}` ⟨*altro*⟩ `}`

dove le ⟨*opzioni*⟩ vengono passate a quella singola onda, e non a tutto il tracciato, ⟨*nome onda*⟩ è il nome dell'onda, mentre ⟨*altro*⟩ dipende dal ⟨*nome d'onda*⟩.

2.1 Onde

Le onde che è possibile disegnare sono:

`p` L'onda di tipo `p`, che richiede anche l'aggiunta della ⟨*polarità*⟩ (valori ammessi `p`, `n`), ⟨*decimi di millivolt picco*⟩ (compresi tra 0.1-0.3 mV) e del tempo in ⟨*millisecondi*⟩.

`p` ⟨*polarità*⟩ `0` ⟨*decimi di milliVolt picco*⟩ ⟨*millisecondi*⟩

Per le onde bifasiche di polarità `d` o `b` si aggiunge anche il secondo picco, come

`p` ⟨*polarità*⟩ ⟨*decimi di milliVolt 1º picco*⟩ ⟨*decimi di milliVolt 2º picco*⟩ ⟨*millisecondi*⟩

`q,r,s` Onde per disegnare il complesso QRS. Tutte e tre prendono come primo argomento la loro altezza in milliVolt e come secondo argomento la loro durata in millisecondi.

`q/r/s` `{`⟨*altezza picco Q/R/S*⟩`}` ⟨*millisecondi*⟩

`i` Onda isoelettrica, prende un solo argomento, che è il tempo in ⟨*millisecondi*⟩.

`i` ⟨*millisecondi*⟩

`t` Ques'onda prende come primo argomento la ⟨*polarità*⟩, che può essere positiva `p` o negativa `n`, come secondo i ⟨*decimi di milliVolt picco*⟩, come opzionale la ⟨*correzione*⟩ in millisecondi nel caso il picco non sia simmetrico e come ultimo i ⟨*millisecondi*⟩.

`t` ⟨*polarità*⟩ ⟨*decimi di millivolt picco*⟩ `[`⟨*correzione*⟩`]` ⟨*millisecondi*⟩

`!` Permette di inserire un picco definito tramite il comando `\newECG`.

`!` ⟨*nome picco*⟩

`?` Permette di inserire un etichetta nel tracciato, alla sinistra dello stesso. Accetta un comando opzionale (di default impostato a `1 cm`), che regola lo spostamento orizzontale.

`?` `[`⟨*spostamento orizzontale*⟩`]` ⟨*testo*⟩

Automaticamente il comando azzera il nodo di partenza a (0,0), permettendo di creare un nuovo tracciato ECG subito sotto.

2.2 Opzioni possibili

2.2.1 Griglia

L'ambiente accetta quattro opzioni utilizzabili per modificare la dimensione della griglia.

`grid top` Accetta una dimensione come valore. Esso indica di quanto viene allargata verso l'alto la griglia rispetto alla dimensione corrente del diagramma.

`grid bottom` Simile a `grid top`, ma allarga la griglia verso il basso.

`grid left` Simile a `grid top`, ma allarga la griglia verso sinistra.

`grid right` Simile a `grid top`, ma allarga la griglia verso destra.

`grid border` L'opzione `grid border =` ⟨*valore*⟩ imposta lo stesso ⟨*valore*⟩ contemopraneamente a tutte e quattro le chiavi.

2.3 Andare a capo

Talvolta alcuni tracciati ECG possono essere troppo lunghi e sforare oltre la gabbia del testo. Per questo è stata ideata la chiave `breaklines`.

`breaklines` `breakindent` Questa chiave permette di abilitare una rottura automatica del tracciato su due linee differenti nel caso in cui esso sfori oltre la gabbia del testo. L'elemento viene poi automaticamente indentato del valore `breakindent`, di default posto a 1 cm. Accetta anche, oltre a `true` e `false`, il valore `arrow` che permette di stampare una freccia come indicatore che il tracciato è stato mandato a capo.

2.4 Titolo del ECG

`ECG title` `ECG title align` È possibile inserire un titolo all'elettrocardiogramma tramite la chiave `ECG title`. L'allineamento del titolo è regolato tramite la chiave `EGC title align`, di cui sono possibili i valori `right`, `center` e `left` (quest'ultimo il valore di default).

2.5 Titolo dei tracciati

2.6 Database delle onde

Per aggiungere un'onda personalizzata occorre utilizzare il comando

`\newECG {`⟨*nome onda*⟩`} {`⟨*codice onda*⟩`}`

L'argomento ⟨*nome onda*⟩ è la stringa che viene poi utlizzata per accedere al ⟨*codice onda*⟩ quando richiamata tramite la chiave `!`.

Il comando controlla se l'onda è già stato definita: in tal caso ignora il codice e stampa un avvertimento.

3 Codice commentato

Prima carichiamo i pacchetti TikZ e xparse.

```
\RequirePackage{tikz}
\RequirePackage{xparse}

\usetikzlibrary{calc}
```

Questo è necessario per definire un nuovo livello. pgf può gestire più livelli, e questo risulta comodo per sovrappore oggetti collocandoli su livelli diversi. Nel caso definisco il livello `sk@back` e lo colloco dietro a quello principale (qui indicato da `main`).

```
\pgfdeclarelayer{sk@back}
\pgfsetlayers{sk@back,main}
\ExplSyntaxOn
```

Definisco alcune chiavi da passare all'ambiente `ecg`. Nello specifico permettono di gestire quanto la griglia sottostante sborda rispetto ai tracciati. grid border imposta lo stesso valore a tutti i lati (di default .2 cm) mentre le altre chiavi impostano lo spessore del rispettivo lato. Il booleano `\sk_ECG_breaklines_bool` serve per gestire i tracciati, abilitando o disabilitando la cesura automatica a fine riga.

```
\bool_new:N \sk_ECG_breaklines_bool
\bool_new:N \sk_ECG_breaklines_arrow_bool

\tikzset {
  breakindent / .store ~ in = \sk_ECG_breakindent_tl ,
  breakindent = 1cm,
  breaklines / .is ~ choice,
  breaklines / true / .code = \bool_set_true:N \sk_ECG_breaklines_bool ,
  breaklines / false / .code = \bool_set_false:N \sk_ECG_breaklines_bool ,
  breaklines / arrow / .code =
   \bool_set_true:N \sk_ECG_breaklines_bool
   \bool_set_true:N \sk_ECG_breaklines_arrow_bool ,
  breaklines / no ~ indent /.code =
   \bool_set_true:N \sk_ECG_breaklines_bool
   \tl_set:Nn \sk_ECG_breakindent_tl { 0 } ,
  breaklines / .default = true,
  breaklines = false,
  ECG ~ title / .store ~ in = \sk_ECG_title_tl,
  ECG ~ title  = {},
  ECG ~ title ~ align / .is~choice ,
  ECG ~ title ~ align / center / .code = \tl_set:Nn \sk_ECG_title_align_tl { north } \tl_set:Nn \sk_
  ECG ~ title ~ align / right / .code = \tl_set:Nn \sk_ECG_title_align_tl { north ~ east } \tl_set:Nn
  ECG ~ title ~ align / left / .code = \tl_set:Nn \sk_ECG_title_align_tl { north ~ west } \tl_set:Nn
  ECG ~ title ~ align = left ,
  grid ~ border / .code =
    \tl_set:Nn \sk_ECG_gridtp_tl { #1 }
    \tl_set:Nn \sk_ECG_gridlf_tl { #1 }
    \tl_set:Nn \sk_ECG_gridrg_tl { #1 }
    \tl_set:Nn \sk_ECG_gridbt_tl { #1 },
  grid ~ top / .store ~ in = \sk_ECG_gridtp_tl,
  grid ~ bottom / .store ~ in = \sk_ECG_gridbt_tl,
  grid ~ left / .store ~ in = \sk_ECG_gridlf_tl,
  grid ~ right / .store ~ in = \sk_ECG_gridrg_tl,
  grid ~ border = .2cm,
}
```

Nuovo booleano che permette di comprendere se siamo alla prima onda o no. Di base impostato su falso.

```
\bool_new:N \sk_ECG_start_bool
```

Viene definito il comando `\ECG` che prende due argomenti: uno opzionale `O{}` (che di default non ha nulla) e uno obbligatorio m. L'opzionale rappresenta le opzioni di TikZpassate poi al comando `\draw[]` (come i comandi per il colore). L'obbligatorio invece è la lista di onde che va poi processata

```
\DeclareDocumentCommand \ECG { O{} m }
  {
  \begin{tikzpicture}
  \coordinate(sk_end) at (0,0);
  \sk_ECG:nn { #1 } { #2 }
  \end{tikzpicture}
  }
```

Sempre analogo del comando `\ECG`, ma pensato per funzionare dentro all'ambiente ecg. Questa versioni infatti non apre ne chiude l'ambiente tikzpicture. Inoltre, oltre ai due precedenti comandi, si introduce il comando facoltativo racchiuso tra parentesi tonde, che permette di collocare il traccaito all'altezza voluta. Il sistema con `\sk_ECG_yshift_dim` toglie per ogni comando `\ECG` 2.5 cm in modo che il successivo comando (se non viene specificato un valore tra parentesi) sia collocato sotto il precedente.

```
\cs_set_nopar:Npn \sk_ECG_env:nnn #1 #2 #3
{
  \tl_if_empty:nTF { #2 }
      { \coordinate(sk_end) at (0,\sk_ECG_yshift_dim);  }
      { \coordinate(sk_end) at (#2); }
  \sk_ECG:nn { #1 } { #3 }
  \dim_set:Nn \sk_ECG_yshift_dim { \sk_ECG_yshift_dim - 2.5cm }
}
```

Definisco la dimensione per lo spostamento in verticale dei vari tracciati. Inoltre definisco due contatori per la numerazione di tutte le onde, uno per il tracciato e uno per l'onda.

```
\dim_new:N \sk_ECG_yshift_dim
\dim_new:N \sk_ECG_x_dim
\int_new:N \sk_ECG_row_int
\int_new:N \sk_ECG_wave_int
```

Ambiente `ecg`. Il suo scopo è permettere di disegnare più tracciati, e collocarci dietro una griglia millimetrata. Inanzitutto azzero il valore dell'altezza per partire nuovamente da zero (non sarebbe necessario, ma per scrupolo è meglio azzerare comunque). Imposto poi che il comando `\ECG` accetti un argomento opzionale tra parentesi quadre, uno opzionale tra tonde e uno obbligatorio. Passo tutto a `\sk_ECG_env:nnn`. Apro l'ambiente `tikzpicture`. La versione asteriscata non utilizza l'ambiente `center`.

```
\DeclareDocumentEnvironment { ecg } { O{} }
{
  \int_zero:N \sk_ECG_row_int
  \int_zero:N \sk_ECG_wave_int
  \dim_zero:N \sk_ECG_yshift_dim
  \DeclareDocumentCommand \ECG { O{} D(){} m }
  {
    \dim_zero:N \sk_ECG_x_dim
```

```
    \int_incr:N \sk_ECG_row_int
    \sk_ECG_env:nnn { ##1 } { ##2 } { ##3 }
  }
  \begin{center}
  \begin{tikzpicture}[#1]
}
```

L'ambiente `pgfonlayer` permette di selezionare su quale livello porre la nostra griglia millimetrata. Affinché si adatti automaticamente ai tracciati sfrutto un nodo di TikZ speciale: current bounding box, le cui dimensioni sono proprio le attuali dimensioni del disegno TikZ. Sfrutto quindi due ancore del nodo `current bounding box`: quella in altro a destra (ancora `north east`) e in basso a sinistra (ancora `south west`). Le chiavi `xshift` e `yshift` permettono di traslare il punto dei valori passati (quelli impostati con le varie chiavi `border`). In tal modo sono definiti due nodi `sk_start` e `sk_stop` che definiscono i vertici in alto a destra e in basso a sinistra della griglia, che sarà semplicemente disegnata con grid e i settaggi desiderati (colore e step). Il tutto, grazie all'ambiente `pfgonlayer`, posto automaticamente sullo sfondo.

```
{
  \tl_if_empty:NF \sk_ECG_title_tl
      { \node at (current ~ bounding ~ box.\sk_ECG_title_align_tl) [inner ~ sep=\c_zero_dim,anchor=\s
  \begin{pgfonlayer}{sk@back}
    \coordinate(grid_start) at ([xshift=-\sk_ECG_gridlf_tl,
      yshift=-\sk_ECG_gridbt_tl] current ~ bounding ~ box. south ~ west);
    \coordinate(grid_stop) at ([xshift=\sk_ECG_gridrg_tl,
      yshift=\sk_ECG_gridtp_tl] current ~ bounding ~ box.north ~ east);
    \draw[color=red!20,step=0.1cm,very ~ thin]
      (grid_start) grid (grid_stop);
    \draw [color=red!50,step=.5cm](grid_start)grid(grid_stop);
    \end{pgfonlayer}
  \end{tikzpicture}
  \end{center}
}

% L'ambiente \env{ecg*} è definito in modo simile a \env{ecg}, ma non allinea al centro il grafico.
%    \begin{macrocode}
\DeclareDocumentEnvironment { ecg* } { O{} }
{
  \int_zero:N \sk:ECG_row_int
  \int_zero:N \sk:ECG_wave_int
  \dim_zero:N \sk_ECG_yshift_dim
  \DeclareDocumentCommand \ECG { O{} D(){} m }
  {
    \dim_zero:N \sk_ECG_x_dim
    \int_incr:N \sk_ECG_row_int
    \sk_ECG_env:nnn { ##1 } { ##2 } { ##3 }
  }
  \begin{tikzpicture}[#1]
}
{
  \tl_if_empty:NF \sk_ECG_title_tl
    { \node at (current ~ bounding ~ box.\sk_ECG_title_align_tl) {\sk_ECG_title_tl}; }
  \begin{pgfonlayer}{sk@back}
    \coordinate(grid_start) at ([xshift=-\sk_ECG_gridlf_tl,
      yshift=-\sk_ECG_gridbt_tl] current ~ bounding ~ box. south ~ west);
```

```
    \coordinate(grid_stop) at ([xshift=\sk_ECG_gridrg_tl,
      yshift=\sk_ECG_gridtp_tl] current ~ bounding ~ box.north ~ east);
    \draw[color=red!20,step=0.1cm,very ~ thin]
      (grid_start) grid (grid_stop);
    \draw [color=red!50,step=.5cm](grid_start)grid(grid_stop);
    \end{pgfonlayer}
  \end{tikzpicture}
}
```

Definisce una nuova sequence. È un costrutto speciale di LaTeX3 che permette di gestire dei dati.

```
\seq_new:N \sk_ECG_seq
```

Dopo aver svuotato la token listi delle onde e la sequence (e settato a vero il booleano di onda iniziale) si suddivide la lista di onde rispetto alle virgole (il secondo argomento di `\seq_set_split:Nnn`) e si salva nella sequence. Il comando `\seq_map_inline:Nn` passa poi ogni singolo elemento della sequence al suo argomento come `##1` (l'argomento `#1` sono le opzioni TikZ).

```
\cs_set_nopar:Npn \sk_ECG:nn #1 #2
{
  \tl_clear:N \sk_ECG_onda_tl
  \seq_clear:N \sk_ECG_seq
  \bool_set_true:N \sk_ECG_start_bool
  \seq_set_split:Nnn \sk_ECG_seq { , } { #2 }
  \seq_map_inline:Nn \sk_ECG_seq { \sk_ECG_use:w ##1 \q_nil { #1 } }
}
```

Macro che processa ogni singola onda: prende 4 argomenti.

1. Il primo opzionale sono eventuali opzioni per una singola onda (pensato principalemente per colorare una singola onda, magari per evidenziarla). Viene passato insieme a `#4` (che rappresenta le opzioni globali del tracciato), alle opzioni di `\draw`, che è salvato nella token list `\sk_ECG_onda_tl`. Il comando `\draw` è impostato con le chiavi osservabili. Lo shift pone il punto zero nell'ultimo node `sk_end`. Al termine, prima di espandere la token list (e rendere effettivo il disegno TikZ), si aggiorna la posizione del nodo `sk_end` e si chiude `\draw` con il punto e virgola necessario.

2. Il secondo è la prima lettera della sigla. Viene passato al comando `\str_case:nn` che lo confronta con tutte quelle presenti e usa quella che coincide con il codice corrente.

3. Il terzo è un argomento delimitato, ossia anche se non sono presenti tonde, il comando raccoglie tutto ciò che trova fino al delimitatore (in questo caso il mark `\q_nil`). Permette di scrivere indipendentemente costrutti come `!{nome}` o `!nome` e ottenere il medesimo risultato. Il comando viene poi passato alle macro interne selezionate dalla prima lettera dell'onda.

```
\DeclareDocumentCommand \sk_ECG_use:w { O{} m u\q_nil m }
{
  \int_incr:N \sk_ECG_wave_int
  \tl_set:Nn \sk_ECG_onda_tl
    { \draw[thick,rounded ~ corners=0.25mm,line ~ cap=round, shift=(sk_end),
    #4,#1] }
  \str_case:nn { #2 }
```

```
    {
      { ! } { \tl_put_right:Nx \sk_ECG_onda_tl
        {
          \cs_if_exist_use:cF { sk_ECG_ #3 } { (0,0) }
        } }
      { ? } { \sk_ECG_onda_label:w #3 \q_nil }
      { i } { \sk_ECG_onda_iso:w #3 \q_nil }
      { p } { \sk_ECG_onda_p:nnnw #3 \q_nil }
      { q } { \sk_ECG_onda_q:nw #3 \q_nil }
      { r } { \sk_ECG_onda_r:nw #3 \q_nil }
      { s } { \sk_ECG_onda_s:nw #3 \q_nil }
      { t } { \sk_ECG_onda_t:nw #3 \q_nil }
      { T } { \tl_put_right:Nx \sk_ECG_onda_tl
        { (0,0) -- (.1cm,0) -- (.1cm,#3*1cm) -- (.5cm,#3*1cm) -- (.5,0)
        -- (1,0) } }
      { * } { \int_decr:N \sk_ECG_wave_int \sk_ECG_onda_ripeti:nw #3 \q_nil { #4 }  \tracingmacros=0
    }
  \tl_put_right:Nn \sk_ECG_onda_tl { coordinate (sk_end)
    let \p1=(current ~ path ~ bounding ~ box.south ~ west),
    \p2=(current ~ path ~ bounding ~ box.north ~ east)
    in ~
    node (ecg-\int_use:N \sk_ECG_row_int-\int_use:N \sk_ECG_wave_int)
       [at=(current ~ path ~ bounding ~ box),text ~ width=\x2-\x1,text ~ height=\y2-\y1]{}
   ; }

  \str_if_eq:nnF { #2 } { * } { \sk_ECG_onda_tl }
  \bool_if:NT \sk_ECG_breaklines_bool
    { \pgfgetlastxy{\sk_ECG_lastx_tl}{\sk_ECG_lasty_tl}
  \dim_set:Nn \l_tmpa_dim
    { \textwidth - \sk_ECG_gridlf_tl - \sk_ECG_gridrg_tl - \sk_ECG_lastx_tl }
  \dim_compare:nT { \l_tmpa_dim < 1.5cm }
    {
      \dim_zero:N \sk_ECG_x_dim
      \dim_set:Nn \sk_ECG_yshift_dim { \sk_ECG_yshift_dim - 2.5cm }
      \coordinate(sk_end) at (\sk_ECG_breakindent_tl,\sk_ECG_yshift_dim);
      \bool_if:NT \sk_ECG_breaklines_arrow_bool {
      \draw[thick,<-,rounded ~ corners]([xshift=-.25cm]sk_end) -| ++ (-.5cm,1.5cm); }
    } }
  \bool_if:NT \sk_ECG_start_bool { \bool_set_false:N \sk_ECG_start_bool }
}
```

Nel caso in cui il primo elemento sia `*`, vengono poi cercati due argomenti: il primo è il numero di ripetizioni, il secondo la lista di onde. Il comando `\prg_replicate:nn` si occupa di ripetere per `#1` volte il comando `\sk_ECG:nn{#3}{#2}`, dove `#3` sono le opioni globali del tracciato, e `#2` la lista di onde da ripetere

```
\cs_set_nopar:Npn \sk_ECG_onda_ripeti:nw #1 #2 \q_nil #3
{
\group_begin:
  \prg_replicate:nn { #1 } { \sk_ECG:nn { #3 } { #2 } }
\group_end:
}
```

Se il primo elemento è `?`, si cerca un argomento opzionale (di default posto a .5 cm) e poi uno delimitato (il testo dell'etichetta). L'etichetta viene quindi salvata nella token list e posta al punto `(#1,0)` e l'allineamento del nodo deciso in base al booleano

`\sk_ECG_start_bool`: se infatti è vero (siamo all'inizio del tracciato) l'etichetta è allineata a sinistra, altrimenti a destra.

`#2` Decide quale derivazione usare in base alla sigla e nell'eventualità non sia prevista, viene stampato come appare. L'utilizzo dei nodi `sk_deriv` serve per posizionare correttamente il tracciato successivo e impedire la sovrapposizione di questo all'etichetta. Inoltre esso permette lo spostamento automatico del tracciato (tiene conto infatti della larghezza dell'etichetta).

```
\DeclareDocumentCommand \sk_ECG_onda_label:w { O{.5cm} u\q_nil }
{
  \tl_put_right:Nx \sk_ECG_onda_tl { (#1,0) node [
  \bool_if:NTF \sk_ECG_start_bool
   { left }
   { right }
  ,name=sk_deriv]
  { \str_case:nnF { #2 }
  {
   { d1 } { I }
   { d2 } { II }
   { d3 } { III }
   { vr } { aVR }
   { vl } { aVL }
   { vf } { aVF }
   { v1 } { V1 }
   { v2 } { V2 }
   { v3 } { V3 }
   { v4 } { V4 }
   { v5 } { V5 }
   { v6 } { V6 }
  } { #2 } } ; \exp_not:N \path(sk_deriv.east) }
}
```

Macro associata alla lettera `i`. L'argomento `#1` viene controllato (se uguale a `k` viene sostituito da `1000`) e poi usato per la lunghezza del tratto.

```
\cs_set_nopar:Npn \sk_ECG_onda_iso:w #1 \q_nil
{
  \tl_put_right:Nx \sk_ECG_onda_tl
  { (0,0) -- ( 0.025 mm *
    \str_if_eq:nnTF { k } { #1 }
      { 1000 }
      { #1 }
   ,0) }
}
```

Onda Q, il primo argomento è l'altezza, il secondo la durata. Sono moltiplicati per i fattori correttivi affinché corrispondano con i millisecondi e i millimetri.

```
\cs_set_nopar:Npn \sk_ECG_onda_q:nw #1 #2 \q_nil
{
  \tl_put_right:Nn \sk_ECG_onda_tl {
  (0,0)--(#1*.0125cm,-#2*.1cm)--(#1*.025cm,0) }
}
```

Onda R: come l'onda Q.

```
\cs_set_nopar:Npn \sk_ECG_onda_r:nw #1 #2 \q_nil
{
  \tl_put_right:Nn \sk_ECG_onda_tl
```

```
    { (0,0) -- (#1*.0125cm,#2*.1) -- (#1*.025cm,0) }
}
```

Onda S: come l'onda Q.

```
\cs_set_nopar:Npn \sk_ECG_onda_s:nw #1 #2 \q_nil
{
  \tl_put_right:Nn \sk_ECG_onda_tl
    { (0,0)--(#1*0.0125cm-0.005cm,-#2*.1cm)--(#1*0.025cm,0) }
}
```

Onda T. `#1` indica se è positiva o negativa (dobbiamo aggiungere la difasica). `#2` sono i millisecondi e `#3` i millimetri.

```
\cs_set_nopar:Npn \sk_ECG_onda_t:nw #1 #2 #3\q_nil
{
  \str_case:nn { #1 }
  {
    { p }
    { \tl_put_right:Nn \sk_ECG_onda_tl
      { (0,0) .. controls (#2*.0625cm,.05cm) and (#2*.075cm,#3*.065cm)
      .. (#2*.112cm,#3*.1cm) -- (#2*.138cm,#3*.09cm) .. controls
      (#2*.175cm,#3*.065cm) and (#2*.188cm,.05) .. (#2*.25cm,0) }
    }
    { n }
    { \tl_put_right:Nn \sk_ECG_onda_tl
      { (0,0) .. controls (#2*.0625cm,-.05cm) and (#2*.075cm,-#3*.065cm) ..
      (#2*.112cm,-#3*.1cm) -- (#2*.138cm,-#3*.09cm) ..
      controls (#2*.175cm,-#3*.065cm) and (#2*.188cm,-.05) .. (#2*.25cm,0) }
    }
  }
}
```

Onda P. In modo simile all'onda T, `#1` distingue tra le varie `p`, `n`, `d` e `b`. `#2` e `#3` sono le altezze dei due picchi, `#4` la durata in millisecondi.

```
\cs_set_nopar:Npn \sk_ECG_onda_p:nnnw #1 #2 #3 #4 \q_nil
{
  \tl_put_right:Nx \sk_ECG_onda_tl { \str_case:nn { #1 }
  {
    { p } { (0,0) -- (.0005*#4,#2#3*.06) -- (.0011*#4,#2#3*.1) --
    (.002*#4,#2#3*.06) -- (.0025*#4,.0) }
    { n } { (0,0) -- (.0005*#4,-#2#3*.06) -- (.0011*#4,-#2#3*.1) --
    (.002*#4,-#2#3*.06) -- (.0025*#4,.0) }
    { d } { (0,0) -- (.01cm,.005cm)
    --  (#4*0.3333-.01cm,#2*.1cm-.008cm)
    -- (#4*0.3333+.01cm,#2*.1cm-.008cm)
    --  (#4*0.6667-.01cm,-#3*.1cm+.008cm)
    -- (#4*0.6667+.01cm,-#3*.1cm+.008cm)
    -- (#4-.01cm,.005cm)
    -- (#4,0) }
    { b } { (0,0) -- (.01cm,.005cm)
    --  (#4*1/3-.01cm,#2*0.045cm+.105cm)
    -- (#4*1/3+.01cm,#2*0.045cm+.105cm)
    -- (#4*1/2-.01cm,.1cm)
    -- (#4*1/2+.01cm,.1cm)
    -- (#4*2/3-.01cm,#3*0.045cm+.105cm)
    -- (#4*2/3+.01cm,#3*0.045cm+.105cm)
    -- (#4-.01cm,.005cm)
```

```
    -- (#4,0) }
  } }
}
```

Il comando `\msg_new:nnn` definisce un messaggio da stampare nel log. Nel caso specifico serve per avertire che la sigla scelta è già utilizzata per un altra onda.

```
\msg_new:nnn { ECG } { wave ~ defined }
  { wave ~ "#1" ~ already ~ defined. ~ Change ~ name. }
```

Definisco una nuova onda con sigla `#1` e codice `#3` (il `#2` è codice opzionale, lo messo nel caso volessimo aggiungere funzionalità a questo comando). Nel caso di onda già definita stampa un avvertimento e non rinomina la precendete.

```
\cs_set_nopar:Npn \sk_ECG_new:nn #1 #2 #3
{
  \cs_if_exist:cTF { sk_ECG_ #1 }
    { \msg_warning:nnn { ECG } { wave ~ defined } { #1 } }
    { \cs_set_nopar:cpn { sk_ECG_ #1 } { #3 } }
}
```

Interfaccia con l'utente per definire nuove onde nel database. `#1` è obbligatorio e contiene il nome dell'onda, `#2` opzionale (come scritto sopra in previsione di futuri sviluppi) e `#3` è il codice TikZ.

```
\DeclareDocumentCommand \nuovoECG { m O{} m }
{
  \sk_ECG_new:nn { #1 } { #2 } { #3 }
}
```

Messaggio nel caso di archivio non trovato

```
\msg_new:nnn { ECG } { fnotf }
  { File ~ "#1" ~ not ~ found. ~ Please ~ check ~ your ~ installation.}
```

Include l'achivio delle onde se presente, altrimente avverte l'utente che il file è assente.

```
\file_if_exist:nTF { archivio }
  { \file_input:n { archivio } }
  { \msg_warning:nnn { ECG } { fnotf } { archivio } }

\ExplSyntaxOff
```