Ingeniería Electrónica - UTN - FRC
Medidas Electrónicas II

Analizador de Estados Lógicos

Guía de Trabajos Prácticos Nº 4

1. Introducción teórica

El analizador lógico surge como una herramienta para analizar fundamentalmente *sistemas lógicos secuenciales síncronos*, que son los más complejos. Por supuesto, pudiendo medir este caso también podemos cubrir los demás circuitos lógicos, aunque *no es imprescindible* para medirlos. Un sistema síncrono es gobernado por secuencias de estados pre-definidas según *máquinas de estados* o *sofware*, que actúan sobre una determinada tecnología digital o *hardware*. Cada capa entre estos dos niveles de abstracción puede presentar distintos problemas que afecten el funcionamiento del sistema; en consecuencia, el analizador lógico debe ser capaz de verificar desde el nivel de software hasta el nivel de hardware.

En la Fig. 1 se puede observar un diagrama general de un analizador lógico. Este instrumento presenta una cantidad de líneas de entrada, que pueden ser 128, 256, etc., permitiendo la adquisición simultánea de varios canales. Estos canales pueden ser almacenados en una memoria interna, lo cual permite desplegar un registro histórico de tales canales. Dichos canales de un bit se puede agrupar en *buses* de ancho configurable, para su mejor interpretación. Por ejemplo, en la Fig. 1 tenemos dos buses A_1 y A_2. Además se tiene una entrada dedicada de clock *clk* y una entrada de calificador *qual*, de las que se hablará más adelante.

El bloque *probe buffer* tiene como función principal separar el analizador lógico del circuito digital a medir, permitiendo la medición sin efectar el funcionamiento normal de éste. De este bloque parte un bus de datos que son tomados pro el bloque *latch*, comandado por una señal de sincronismo. Los datos capturados por el latch son almacenados en memoria de alta velocidad a través de un bloque *DMA*. Finalmente, los datos almacenados son enviados a un microcontrolador para su presentación en una pantalla.

Aparte del camino de datos, existen bloques que controlan el comportamiento del instrumento; en términos generales éstos son:

1. **Lógica de Sincronismo:** utiliza la señal de reloj y de calificación a fin de generar *pulsos cualificados o ponderados de escritura.*

2. **Lógica de disparo:** consta de comparadores que permanentemente contrastan el dato que entrega el latch con palabras de disparo definidas pro el usuario (latches de disparo). Cuando se produce una coincidencia, se envían señales a los *secuenciadores y retardadores.*

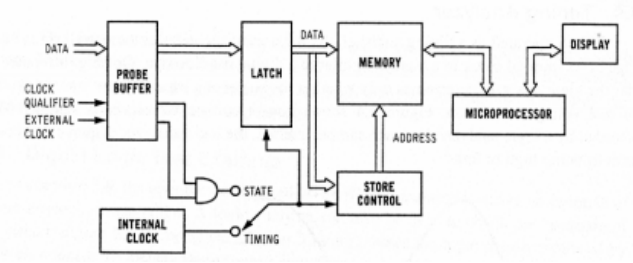


Figura 1: Diagrama en bloque de un Analizador de Estados Lógicos

Sample Number	ADDR	DATA	STAT
	= FFF0 34D8	= XXXX XXXX	= XX XXXX
-8	0000 41B0	0241 23D7	03 23D7
-7	0000 41B1	1F41 23D7	0B 23D7
-6	0000 41B2	FB41 23D7	13 23D7
-5	0000 41B3	5241 23D7	0B 23D7
-1	FFF0 3187	5541 03E7	0B 03E7
3	FFF0 34DB	A641 03E7	0B 03E7
7	FFF0 34DF	7841 03E7	0B 03E7
11	FFF0 34E3	E841 03E7	0B 03E7
15	FFF0 34E7	2941 03E7	0B 03E7
19	FFF0 6A0F	F441 03E7	0B 03E7

Figura 2: Ejemplo de uso de un analizador de estados

3. **Secuenciadores y retardadores:** procesan las señales provenientes de la lógica de disparo y dirigen en consecuencia la operación de lso demás sistemas, brindando la flexibilidad necesaria para implementar los llamados *modos de disparo* del instrumento.

1.1. Operación general

El sistema secuencial a medir cambia al ritmo de un determinado reloj, por lo que se debe tener en cuenta no sólo las líneas de interés sino una línea de clock que indica cuándo se deben muestrear estas líneas. Sobre esta base, el analizador lógico se puede utilizar en dos modos principales:

- Analizador de estados
- Analizador temporal

1.1.1. Analizador de estados

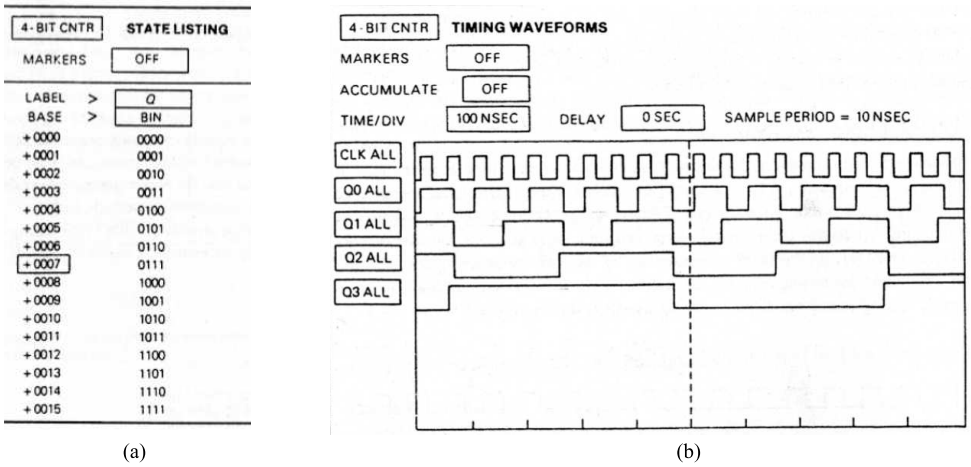
En el modo de análisis de estados se utiliza un *reloj externo* al instrumento, proveniente del circuito bajo ensayo, el cual determina los momentos en que se adquieren las señales. En este modo, el analizador opera en forma sincrónica con el circuito a ensayar. Como reloj se pueden utilizar el reloj principal o cualquier señal proveniente del sistema a ensayar. En el modo de estados no se puede saber qué ocurre *entre flancos del reloj*, para lo cual sería necesaria una tasa de muestreo mayor; a raíz de ello surge el modo de *analizador temporal*.

El analizador en modo de estados puede representar la información en distintos *formatos*. El formato básico para señales digitales es el binario, donde cada línea se encuentra en estado 1 o 0. Cuando se tienen muchas líneas de datos, sin embargo, este formato no es práctico y se reemplaza por otros. El formato *decimal*, con o sin signo, es una alternativa; sin embargo su forma totalmente compacta ocasiona que se pierda totalmente el seguimiento intuitivo de los bits individuales. El formato hexadecimal, por otro lado, es muy conveniente para representar grandes números binarios en forma compacta, permitiendo el seguimiento de los bits individuales ya que éstos se agrupan sólo de a cuatro. Por ejemplo, un número de 32 bits de valor 10111000011110001111101000010010 es 24179188 en decimal, lo que no permite distinguir en absoluto los bits individuales. En el formato hexadecimal, el número se divide de a cuatro bits obteniendo 1011 1000 0111 1000 1111 1010 0001 0010, cuya representación hexadecimal es B878FA12. Esta forma es un compromiso más conveniente de formato, permitiendo el seguimiento intuitivo de los bits individuales. Finalmente, se puede utilizar el código ASCII, diseñado para información alfanumérica y utilizado por gran cantidad de dispositivos digitales.

Se puede configurar el analizador de estados lógicos para que traduzca la información, de esta forma se podría leer más comodamente. Por ejemplo definir que de la linea 0 a la 3 es la dirección, la linea 4 a la 7 es el dato y las últimas cuatro lineas que sean el estado. Como se observa en la figura **??**

1.1.2. Analizador temporal

El modo de análisis temporal se utiliza cuando se desea medir las señales lógicas con una base de tiempo más precisa que el modo de estados. Esta base de tiempo proviene de un *reloj interno*, propio del analizador e independiente del reloj del sistema a medir; por este motivo se dice que la operación



(a) (b)

Figura 3: Modos del analizador lógico: (a) análisis de estados, (b) análisis temporal

del analizador es en este caso *asíncrona*. Este reloj interno posee frecuencia controlable por el usuario y normalmente más alta que la frecuencia de reloj del sistema que se pretende medir.

En el modo de analizador temporal, el analizador lógico opera en forma similar a un osciloscopio de almacenamiento digital, con la importante diferencia de que almacena sólo el estado lógico de la entrada. Es decir, opera como un OAD con ADC de 1 bit.

1.1.3. Comparación de ambos modos

La presentación en pantalla en ambos modos es sustancialmente diferente. En el modo de análisis temporal se presentan *diagramas de onda* (con transiciones ideales), mientras que en el modo de análisis de estados se muestra una *lista de estados* del sistema digital.

Ambos modos de funcionamiento se pueden comparar mediante un caso simple de medición de un contador digital de 4 bits. En la Fig. 3(a) se observa la presentación de este contador de 4 bits en modo de análisis de estados, mientras que la Fig. 3(b) muestra el mismo contador observado en modo de análisis temporal. En el primer caso se presentan dos columnas; la primera de ellas representa el número de muestra respecto del punto de disparo (esto se estudiará luego), mientras que la segunda representa el valor binario del contador en el instante de muestreo. En el segundo caso, en tanto, se representan las líneas de un bus genérico de 8 bits en la forma de odas temporales.

En estos diagramas suponemos que las capturas se realizan en el *flanco ascendente* del reloj de muestreo. Un detalle a destacar es que en el modo temporal se puede capturar el reloj del sistema como una señal más, dado que el reloj de muestreo es de frecuencia más alta. En el modo de estados, en tanto, la captura del reloj del sistema arrojaría un permanente estado en bajo, que es el estado en que se encuentra antes de cada flanco de subida.

Decimos que el modo de estados permite obtener información sobre el software, ya que la mínima transición detectable es un ciclo del clock del sistema.. Si se toma un reloj externo al analizador y se captura en sus flancos ascendentes, se pueden seguir perfectamente los estados del sistema, que obedecen a un determinado flujo de software. Sin embargo, la anomalía en línea de trazos, que es un problema ocasionado por fallas de hardware, pasa desapercibida. Utilizando un clock interno al analizador *Clk_int*, se eleva la tasa de muestreo a voluntad para capturar esta anomalía.

1.1.4. Muestreo transicional

En el muestreo transicional, el sistema de adquisición captura el estado de entrada en al latch y detecta si ocurrió una transición respecto al estado anterior. Sólo en el caso de haber ocurrido una transición, el analizador almacena en memoria el estado y su tiempo relacionado. De esta manera, no se requiere almacenar todos los estados intermedios donde no ocurren transiciones. Se observa fácilmente que, si bien este tipo de muestreo puede ser aplicado a cualquier sistema, *sus mayores beneficios de ahorro de memoria se obtendrán para sistemas con transiciones de estado muy infrecuentes.*

Se observa que en el muestreo común se almacena un estado para cada ciclo de reloj, por lo que a partir del periodo de reloj sabemos a qué momento corresponde cada estado. En el muestreo transicional, en cambio, se debe almacenar el tiempo o múltiplo de periodo de reloj relacionado a cada estado almacenado.

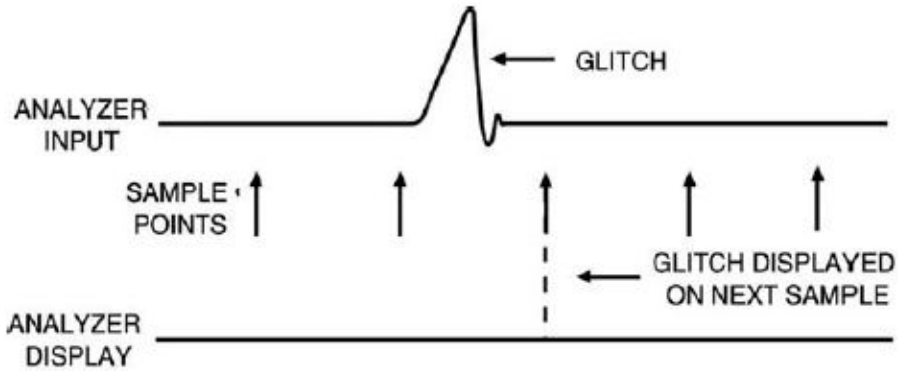


Figura 4: Glitch

De este modo, podemos observar que para sistemas con transiciones muy frecuentes el consumo de memoria puede ser aún mayor que en el muestreo común, ya que se almacenan dos palabras por cada captura.

Como ejemplo se pueden observar las transiciones de una línea lógica en la Fig. **??**(b). En este caso, se observa que sólo cuatro transiciones se producen en un tiempo de diez ciclos de clock, por lo que sólo se almacenan cuatro bits con sus tiempos asociados. Por ejemplo, cuando el analizador encuentra una transición en t_1, almacena el estado 1 con su tiempo relacionado y comienza a contar tiempo. Cuando llega a t_2, almacena el estado 0 y el incremento de tiempo relacionado, reiniciando el conteo de tiempo. Lo mismo sucede en t_3 y t_4.

1.1.5. Glitch

Un glitch se define como cualquier transición que cruce el umbral lógico una o más de una vez entre muestras. Otra forma de definirlo es la de que es un suceso intermitente que produce un error de un circuito a alta velocidad. Para poder ser detectados depende el nivel de umbral que se elija. Una traducción de glitch puese ser falla, entonces estos fallos en los sistemas digitales pueden ser problemáticos. Los glitchs tienen la mala costumbre de aparecer en los momentos más inoportunos con los resultados más desastrosos. Los analizadores lógicos actuales tienen captura de glitch y capacidad de activación que hace que sea fácil rastrear problemas de glitch esquivos. En la figura **??** se observa este caso

1.1.6. Disparo de un analizador lógico

Un palabra que se utiliza mucho cuando se usa un osciloscopio es trigger. Para marcar la diferencia utilizaremos la palabra disparo cuando se trate de un analizador lógico. También a menudo se denomina "punto de rastreo". A diferencia de un osciloscopio que comienza el rastreo justo después del disparador, un analizador lógico captura continuamente los datos y detiene la adquisición después de que se encuentra el punto de rastreo.

Por lo tanto, un analizador lógico puede mostrar información antes del punto de rastreo, que se conoce como tiempo negativo, así como información después del punto de rastreo.

Establecer especificaciones de disparo en un analizador de tiempo es un poco diferente de configurar el nivel de disparo y la pendiente en un osciloscopio. Muchos analizadores activan un patrón de altos y bajos a través de las líneas de entrada. Se le puede especificar al analizador de estados lógicos que empiece a adquirir a partir de una palabra o cuando se active una linea en particular.

Se puede ver en el ejemplo de la figura **??**, que la palabra se eligió para cuando las salidas desde la 0 a la 3 estén activadas o que es lo mismo que decir en alto.

En la configuración también se elige el flanco, si se desea que suceda en el de subida o el de bajada.

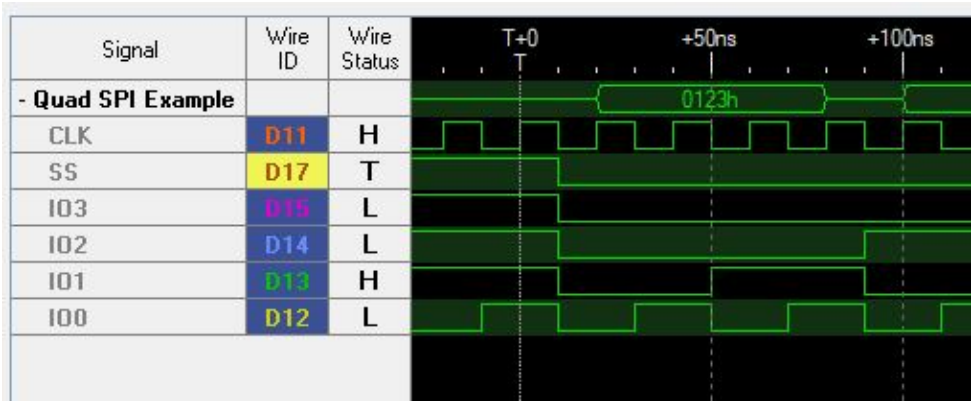


Figura 5: Configuración de palabra de disparo

2. Práctica de Laboratorio

NOTA: ya que se debe compartir el uso de equipos, se recomienda familiarizarse con los puntos principales de los manuales y los conceptos teóricos antes de asistir al laboratorio a fin de no tener que repetir procedimientos.

2.1. Objetivos

- Familiarizarse en el uso de las técnicas para análisis de estados lógicos
- Comprender su relación con las técnicas en el dominio del tiempo y los casos de uso de cada una
- Aplicar el análisis de estados lógicos en el dsPIC utilizado en Técnicas Digitales 3.

2.2. Materiales

Este trabajo práctico se realiza completamente en el Laboratorio de Técnicas Digitales e Informática.

- Analizador de estados lógicos (LA1034 LogicPort)
- KDSP v3.1
- PC

2.3. Procedimiento

Para generar las señales a medir se utilizará la placa de desarrollo KDSP v3.1 y los software vistos en la materia **técnicas digitales 3**. Las señales se generarán mediante un programa simple provisto por esta guía o mediante los ejemplos que el mismo software trae consigo (SPI, I2C, etc), por si se necesita entrar en detalle en la programación de Arduino en la carpeta de *Google Drive* se encuentra un manual de usuario.

Para adquirir las señales se utilizará el software de LogicPort y el instrumento Analizador de estados lógicos (LA1034 LogicPort).

Figura 6: Analizador de estados lógicos (LA1034 LogicPort)

2.4. Ejemplo de Configuración

Para comenzar, seguiremos un ejemplo del software para la calibración de las funciones principales, en el cual veremos como comenzar.

1. Primero abriremos el software *LogicPort* que se encuentra en el escritorio de la PC, con el equipo desconectado nos aparece la opción de observar una Demostración (demo) del programa, elegiremos esta para poder observar las señales.

   En este ejemplo configuraremos las opciones para poder ver:

   - 4 Lineas de datos
   - 4 Lineas de Address
   - Señales de RST, IRQ, CLK, RD, WR
   - Un nivel de tensión CMOS de +5V
   - Un CLK de 25MHz

2. Una vez dentro del programa seleccionaremos $File \rightarrow New$ para generar el archivo de prueba. Una ventana nos solicitará el nombre del archivo y donde guardarlo. Para continuar con el ejemplo seleccionaremos $File \rightarrow Open \rightarrow QuickStart$

3. Primero borraremos las señales preconfiguradas de la pantalla realizando un clic derecho en la zona de señales y haciendo clic sobre *Remove All*, el software para nuestra comodidad nos permite modificar los nombres de las señales, también nos facilita *agrupar* señales para el caso que tengamos varios grupos. Primero modificaremos los nombres, yendo a $Setup \rightarrow SignalNames$ y nombraremos las señales de la siguiente manera:

   - D0 a D3 = Data0 a Data3
   - D4 a D7 = Address0 a Address3
   - D24 = RST
   - D25 = RD
   - D26 = WR
   - D27 = IRQ

   Para agrupar las señales tenemos que entrar en $Setup \rightarrow Groups \rightarrow Create$, armaremos dos grupos, el de Datos y el de Address.

4. Sobre la ventana de señales elegiremos los dos grupos creados y las 4 señales nombradas para poder visualizar en pantalla. Esto lo haremos haciendo clic derecho sobre esta ventana y selecciona *Add Groups* y *Add Signal*.

5. El siguiente paso es configurar el muestreo de las señales, como se puede ver en la figura 5 por defecto el programa viene con *timing mode* que significa que el clock es interno, siendo que el *state mode* nos permite seleccionar uno de los dos cables (o ambos) blancos como Clocks externos. La opción de *Compression* nos permite elegir cuando tomar la información para ahorrar memoria, por lo que solo tomará información desde los cables cuando exista una transición.

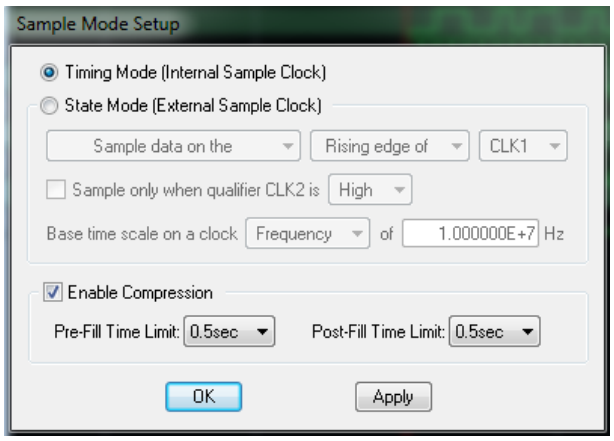


Figura 7: Sample Mode

Luego tenemos tres parámetros para modificar:

a) Sample Rate

b) Logic Threshold

c) Pre-Trigger Buffer

6. El siguiente paso, es uno de los más importantes, ya que elegiremos las condiciones de trigger. A diferencia de los osciloscopios digitales (en los cuales el trigger era un nivel de tensión) en los analizadores de estados logicos el trigger es una *Palabra (Pattern en el software)* o un *Flanco (Edge en el programa)*.

   En el ejemplo que estamos armando nosotros tendriamos una señal de Interrupcion IRQ y una señal de RST, por lo que nuestra condición de trigger sería tomar la señal siempre que suceda una interrupción *siempre y cuando* $RST = 0$.

   Por lo que elegiremos la opción *Trigger When level B is satisfied* y la configuración final sera como se puede observar en la figura 6. Para poder finalizar con la configuración volveremos a la ventana de señales y agregaremos las columnas de *Pattern B* y *Edge B*, esto lo hacemos para poder observar que se pueden elegir de a dos modos de trigger (A y B) simultaneos. Debajo de cada la columna *Edge B*, elegiremos el valor de flanco de subida para IRQ y seleccionaremos en la columna *Pattern B* $RST = 0$.

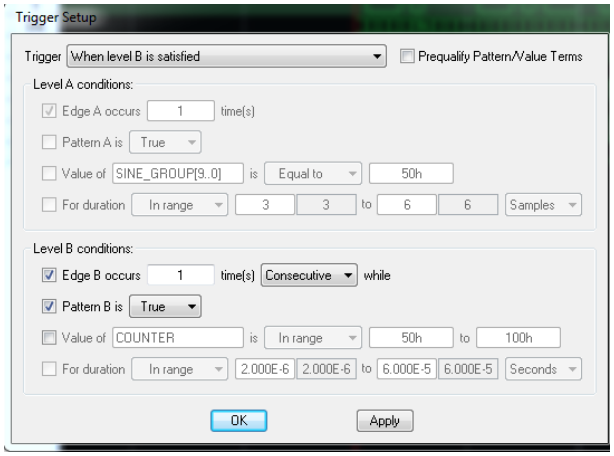


Figura 8: Trigger Mode

7. Ahora veremos el modo de adquisición, siendo 4 las posibilidades

   a) Single, realiza una captura cuando se cumpla la condición de trigger.

   b) Recurring (Recurrente), realiza capturas constantes siempre y cuando se cumpla el *Trigger*.

   c) Halt (Alto), frena la adquisición.

   d) Trigger Immediate, ignora las condiciones de trigger y toma una muestra.

8. De esta forma concluimos con la configuración básica de los distintos comandos del analizador de estados lógicos, vale resaltar que cuando se trata de un demo, no hace caso a las condiciones de trigger.

3. Práctica para la evaluación

El software nos provee varios casos de ejemplos. Los mismos no necesitan de la conexión del hardware para ser utilizados. Los mismos se encuentran en la carpeta $LogicPort\backslash Projects\backslash Examples$. Vale aclarar que como los ejemplos ya fueron medidos, solo se pueden observar y no realizar cambios en los parámetros de adquisición, para poder ver claramente su funcionamiento se recomienda la aplicación del instrumento en conjunto con el dsPIC de la asignatura Técnicas Digitales 3.

Se pueden observar casos muy complejos como el análisis de un controlador de sdram o una simple interpretación digital de una salida de un micro. En estos ejemplos se pueden visualizar la mayoría de comunicaciones digitales existentes.

En las preguntas del aula virtual, se hará incapie sobre las lineas que se necesitan y los parámetros que se utilizaron para cada medición.

Referencias

[1] *LogicPort Logic Analyzer Intro.* Intronix Logic Port.

[2] *Archivo de Help LogicPort Logic Analyzer Intro.* Intronix Logic Port.

[3] *Manual de uso básico dsPIC V3.1*

[4] *Guia Nº 5 Introducción al dsPIC*

[5] *Nota de Aplicación Agilent Nº 1337*

[6] *Logic Analyzer Fundamentals tektronik*