АНАЛИЗ И СИНТЕЗ НА ЛОГИЧЕСКИ СХЕМИ

Лекция

12. РЕГИСТРИ

12.1. Общи сведения и предназначение

Регистърът е последователностна схема. Той е един от типичните представители на възли в цифровата изчислителна машина наред с броячите, дешифраторите и суматорите. Регистрите са тригерни схеми, предназначени за запомняне на n разредни двоични числа и за осъществяване на някои преобразувания с тях. Тригерът може да съхранява само един двоичен разряд. Често обаче възниква необходимост да се съхранява група битове-така наричаната двоична *дума* (binary *word*) – например от 8 бита (1 байт). За съхраняване на тези 8 бита могат да се използват 8 тригера, организирани в 8 разряден *регистър*. Следователно, той представлява съвкупност от тригери, чиито брой съответства на броя на разрядите в думата и притежава спомагателни схеми, осигуряващи изпълнението на основните *микрооперации* в регистрите. Тези спомагателни схеми са съставени от логически елементи, образуващи комбинационна схема. Регистрите се използват още за преобразуване на последователния код в паралелен и обратно, за преместване на определен брой разряди в дясно (ляво), което е необходимо при нормализацията на числата. В регистъра за всеки разряд на съхраняваната дума има един вход и един или два изхода, ако числото трябва да се изведе в прав, обратен или парафазен код. Предвиждат се отделно и входове за установяване на регистъра в първоначално състояние и за преместване на съхраняваната дума. Тези входове се обединяват за всички разряди на думата.

Регистрите са основните елементи в архитектурата на повечето централни процесори. Те представляват малки памети, които са част от блока за управление. Както и клетките на паметта, регистрите се състоят от двоични клетки и притежават адреси, които служат за тяхното идентифициране. Броят на регистрите обаче е много малък. В тях могат да се запомнят данни, докато някоя магистрала или друг блок се освободи за да ги приеме или докато не се извикат от програмата. Регистрите под програмно управление са много удобни, защото централния процесор може да получава данни от тях, без да се обръща към паметта. Регистрите, които не са под програмно управление служат за запомняне на данни за по-късно използване.

По-важни типове регистри са: програмни броячи; регистри на инструкции; адресни регистри; акумулатори; регистри с общо предназначение; индексни регистри; регистри за състояние; указател на стека.

Броят на вътрешните състояния при регистровите схеми обикновено е равен на броя на ЕА (тригерите) в тях.

12.2. Класификация. Основни микрооперации изпълнявани от регистрите

Класификацията на регистрите може да се извърши по различни признаци. В зависимост от начина за въвеждане информацията в регистъра се различават: регистри с предварително изчистване (нулиране) и регистри без предварително нулиране. Според начина за приемане на информацията регистрите са: с приемане на информация в последователен код и с приемане на информация в паралелен код. В зависимост от начина за извеждане на информацията регистрите се разделят на: регистри предаващи информацията в прав, обратен и парафазен код. Регистрите може да се разделят още на регистри без преместване и регистри с преместване. Последните от своя страна се делят на: преместващи регистри в дясно, в ляво и реверсивни (двупосочни) преместващи регистри. При тези преместващи регистри в зависимост от управлението на специални схеми, които осъществяват връзките между тригерите, преместването се извършва в едната (в дясно) или другата посока (в ляво). Според вида на използваната елементна база регистрите се разделят на регистри с импулсно-потенциални елементи, с потенциални и с интегрални логически елементи.

За регистровите схеми са характерни следните особености:

- състоянието на всеки тригер Q_i зависи от състоянието на съответната входна променлива x_i и/или от състоянието на съседните тригери Q_{i+1} и Q_{i-1};

- използваните в даден регистър тригери обикновено са еднотипни с изключение евентуално на двата крайни тригера;

- под действието на управляващи сигнали (допълнителни входни променливи) могат да се извършват следните най-разпространени основни видове *микрооперации*:

- установяване на регистъра в нулево състояние (*нулиране*) – всички тригери се привеждат в състояние нула „0”. Тя се управлява от сигнала на вход $y_{нул}$;

- приемане на информация в регистъра от друг възел или устройство (*запис*) – всеки тригер Q_i приема състоянието (значението) на съответната входна променлива x_i;

- извеждане на информацията към друг възел или устройство (*четене*) – сигналите от правите изходи на тригерите (*прав* код) или от инверсните (при *обратен* код) се подават към изхода на регистъра. Възможен е и вариант на едновременно подаване на правите и инверсните изходи на тригерите към изхода на схемата (*парафазен* код);

В регистрите обикновено се извършват и поразрядни логически операции, а именно:

- логическо събиране (сумиране) - извършва се поразрядно събиране на старото съдържание на регистъра с подаден входен код;

- логическо умножение - извършва се поразрядно логическо умножение на старото съдържание на регистъра с определен входен код;

- поразрядно събиране - поразрядно сумиране по модул 2;
- преместване на думата в ляво или в дясно на определен брой разряди, един или повече;
- преобразуване кода на числата при приемане и извеждане;
- преобразуване на последователен код на думата в паралелен и обратно;
- съхраняване – не се реализира нито една от приведените по-горе микрооперации, регистърът не променя състоянието си.

Изброените до тук схемни особености и основни микрооперации позволяват да се заключи, че синтезът или съответно анализът на един регистър се свежда до разглеждането само на един негов разряд (или на два при микрооперация преместване). За този разряд трябва да се определи възбудителната функция на тригера и изходната му функция. При това за нормална работа на регистъра е необходимо в даден момент да се реализира една единствена микрооперация, т.е. да се подава активен сигнал само на един управляващ вход. Благодарение на това изискване възбудителните функции за всяка микрооперация могат да се синтезират поотделно и след това да се обединят като логическа сума в една обща възбудителна функция.

При регистрите най-целесъобразно е да се използват тригери тип D, RS или JK. Ако във възможностите на регистъра влиза микрооперацията поразрядно двоично събиране по модул 2, известно удобство се постига при използване на тригери тип JK. Използването на тригери тип T в регистрите е нецелесъобразно.

Ще бъдат разгледани някои от изброените микрооперации и синтезирането на схемите за тяхното реализиране с помощта на логически елементи и тригери.

Нулиране-при тази микрооперация всички разреди на регистъра се привеждат в състояние 0. Тя се управлява от сигнала на вход $y_{нул}$;

Запис-при тази микрооперация всеки тригер Q_i приема стойността на съответната входна променлива x_i (таблица 12-1). Микрооперацията се осъществява под управлението на сигнал $y_{зап}$;

$$Q^{t+1} = x_i \; y_{зап} \qquad (12.1)$$

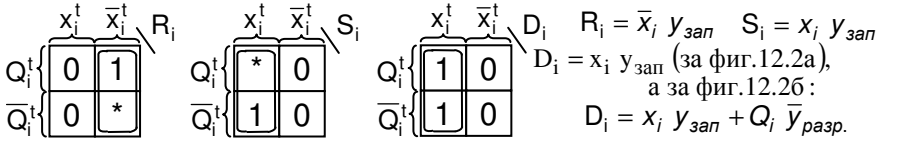
Както за тази, така и за останалите регистрови микрооперации, получаването на възбудителната функция на тригера и изходната му функция се извършва от таблица на преходите, към която са добавени колонки, озаглавени с входовете на използваните тригери (таблица 12-1). Всяка колонка се разглежда като самостоятелна функция, наричана *възбудителна*. Последната се получава след минимизирането ѝ по някой от известните методи за минимизация, но най-често чрез графичен метод с карта на Вейч (Карно).

При получаването на възбудителните функции се използва съответствието между колоните за входовете на тригерите и тези на независимите променливи.

Таблица 12-1

№	x_i^t	Q_i^t	Q_i^{t+1}	R_i	S_i	D_i
0	0	0	0	*	0	0
1	0	1	0	1	0	0
2	1	0	1	0	1	1
3	1	1	1	0	*	1

От таблица 12-1 следва намирането на възбудителните функции, след като те предварително са представени в карти на Вейч и минимизирани.



$R_i = \bar{x}_i \; y_{зап}$ $S_i = x_i \; y_{зап}$

$D_i = x_i \; y_{зап}$ (за фиг.12.2а),

а за фиг.12.2б:

$D_i = x_i \; y_{зап} + Q_i \; \bar{y}_{разр.}$

На фиг. 12.1 и фиг. 12.2 са показани синтезираните схеми за осъществяване на микрооперация *запис* с избраните типове тригери. Записът при схемата с RS тригер е еднотактен (фиг. 12.1), а с D тригер (фиг. 12.2), двутактен ($1^{-ви}$ такт – Clock = 1 ($y_{зап} = 0$), тригерът се нулира; $2^{-ри}$ такт – Clock = 1 ($y_{зап} = 1 - x_i$ се записва в тригера)).

Характерна особеност при използването на тригер тип D (както за тази, така и за останалите микрооперации) е добавянето на члена $Q_i \; \bar{y}_{разр.}$ за гарантиране запазването на старото състояние на тригера при отсъствие на разрешаващ микрооперацията сигнал (фиг. 12.2б).

Четене (извеждане). Единствената микрооперация, която не е свързана с промяна на съдържанието на регистъра. Тя се осъществява в два варианта:

а) четене в *прав* код – при подаване на сигнал 1 на входа $y_{четене}$, сигналът от единичния изход на всеки тригер Q_i се предава на изхода;

б) четене в прав или *обратен* код – в този случай регистърът трябва да има два управляващи входа $y_{четене\ прав}$ ($y_{чп}$) и $y_{четене\ обратен}$ ($y_{чо}$); на изхода се получава при сигнал 1 на вход $y_{чп}$ сигнал от единичния изход Q_i, а при сигнал 1 на вход $y_{чо}$ – от нулевия изход $\overline{Q_i}$. Сигналите от правите и инверсните изходи на тригерите едновременно се подават към изхода на схемата при четене в *парафазен* код (трети вариант).

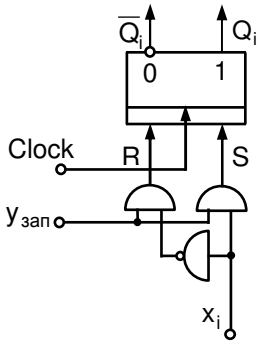
Поразрядно логическо събиране (дизюнкция). При подаване на сигнал 1 на входа $y_\vee$ състоянието на всеки тригер от регистъра се определя от зависимостта (12.2)

$$Q_i^{t+1} = (Q_i^t + x_i^t) y_\vee , \qquad (12.2)$$

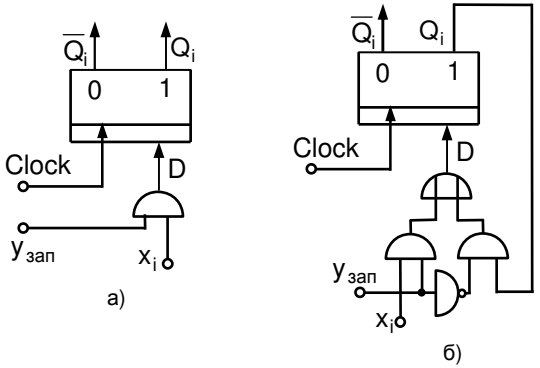
т.е. новото съдържание на регистъра е поразрядна логическа сума от старото му съдържание и подадената на входовете му дума.

Поразрядно логическо умножение (конюнкция). При подаване на сигнал 1 на входа $y_\wedge$ състоянието на всеки тригер от регистъра се определя от

зависимостта (12.3)



Фиг. 12.1. Схема за еднотактен запис с RS тригер



Фиг. 12.2. Схема за двутактен запис с D тригер без а) и с б) гарантиране запазването на старото състояние на тригера при отсъствие на разрешаващ микрооперцията сигнал

$$Q_i^{t+1} = Q_i^t x_i^t y_{\wedge}, \qquad (12.3)$$

т.е. новото съдържание на регистъра е поразрядно логическо произведение от старото му съдържание и подадената на входовете му дума.

Поразрядно двоично събиране по модул 2. При подаване на сигнал 1 на вход $y_{\oplus}$ състоянието на всеки тригер от регистъра се определя от зависимостта (12.4)

$$Q_i^{t+1} = (Q_i^t \oplus x_i^t) y_{\oplus}, \qquad (12.4)$$

т.е. новото съдържание на регистъра представлява резултат от двоичното събиране по модул 2 между старото му съдържание и подадената на входовете му дума.

Преместване на един или няколко разряда вляво. Съдържанието на регистъра се премества под действие на сигнал 1 на входа $y_{пл}$, като всеки тригер приема състоянието на съседния си отдясно, (12.5) т.е.

$$Q_i^{t+1} = Q_{i+1}^t \, y_{пл}. \qquad (12.5)$$

При това най-десният тригер може да не измени състоянието си ($Q_n^{t+1} = Q_n^t \, y_{пл}$), да се нулира ($Q_n^{t+1} = 0$), да приеме стойността на входната променлива x_n ($Q_n^{t+1} = x_n^t \, y_{пл}$) или да приеме състоянието на най-левия тригер от регистъра ($Q_n^{t+1} = Q_1^t \, y_{пл}$). В последния случай преместването се нарича *циклично*, а самият регистър се нарича *кръгов* (*цикличен*) *преместващ регистър*.

Преместване на един или няколко разряда вдясно. Съдържанието на регистъра се премества под действие на сигнал 1 на входа $y_{пд}$, като всеки тригер приема състоянието на съседния си отляво, (12.6) т.е.

$$Q_i^{t+1} = Q_{i-1}^t \, y_{пд}. \qquad (12.6)$$

Най-левият тригер може да не промени състоянието си ($Q_1^{t+1} = Q_1^t \, y_{пд}$), да се нулира ($Q_1^{t+1} = 0$), да приеме стойността на входната променлива x_1 ($Q_1^{t+1} = x_1^t \, y_{пд}$) или да приеме състоянието на най-десния тригер от регистъра ($Q_1^{t+1} = Q_n^t \, y_{пд}$) (*циклично преместване вдясно*).

Съществуват регистри, в които може да се осъществи преместване вляво или вдясно и на повече от един разряд.

Съхраняване. Ако не се реализира нито една от приведените по-горе микрооперации, т.е. $y_{нул} = y_{зап} = y_{\vee} = y_{\wedge} = y_{\oplus} = y_{пл} = y_{пд} = 0$, регистърът не изменя съдържанието си, т.е. всеки тригер запазва старото си състояние (12.7)

$$Q_i^{t+1} = Q_i^t \, \overline{y}_{нул} \, \overline{y}_{зап} \, \overline{y}_{\vee} \, \overline{y}_{\wedge} \, \overline{y}_{\oplus} \, \overline{y}_{пл} \, \overline{y}_{пд}. \qquad (12.7)$$

В съответствие с изложеното дотук функцията на преходите на $i^{-ия}$ разред на регистъра, който може да осъществява всички изброени микрооперации, ще има вида (12.8)

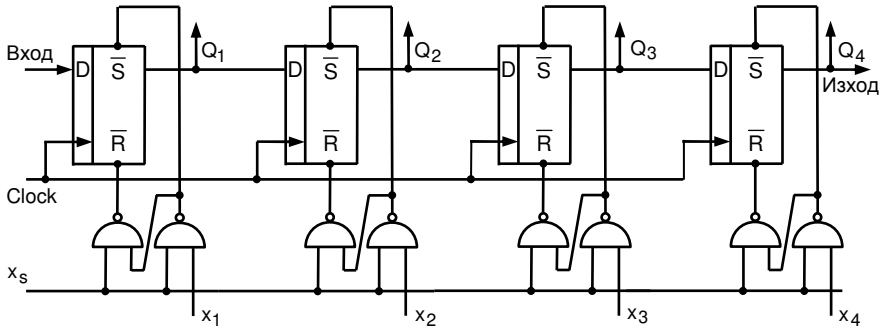
$$Q_i^{t+1} = \overline{y}_{нул}(y_{зап} x_i^t + y_{\vee}(Q_i^t + x_i^t) + y_{\wedge} Q_i^t x_i^t + y_{\oplus}(Q_i^t \oplus x_i^t) + y_{пл} Q_{i+1}^t + y_{пд} Q_{i-1}^t + \overline{y}_{зап} \, \overline{y}_{\vee} \, \overline{y}_{\wedge} \, \overline{y}_{\oplus} \, \overline{y}_{пл} \, \overline{y}_{пд} Q_i^t). \qquad (12.8)$$

Получаването за всяка от изброените по-горе микрооперации функция на преходите, става по аналогичен на показания за микрооперация запис начин (таблица 12-1). Това се отнася и за получаването на възбудителните функции.

12.3. Преместващи регистри

Това са последователни (серийни) регистри, които освен за кратковременно съхраняване на двоичната информация се използват и за преместването ѝ на определен брой разряди наляво или надясно. С малки изключения тези регистри са синхронни, т.е. преместването, приемането и предаването на информацията става под действието на тактови сигнали. За реализацията им най-често се използват D или JK тригери. От разгледаните микрооперации *преместване* и *запис* следва примерната схема на такъв четириразряден регистър (фиг. 12.3).

Регистърът има един последователен вход (D входът на първия регистър) и един последователен изход (изходът на четвъртия регистър Q_4). При първия тактов сигнал $1^{-ият}$ тригер се установява в състояние съответстващо на това на последователния вход. При втория тактов сигнал $2^{-ият}$ тригер приема състоянието на първия, а той от своя страна – новото състояние на последователния вход и т.н. Реализира се *преместване* на входното двоично число отляво надясно.



Фиг. 12.3. Четириразряден преместващ регистър

Освен последователен вход преместващите регистри имат и паралелни входове (x_1, x_2, x_3, и x_4) за запис на двоична информация в паралелен код под действието на сигнал за запис x_s. Освен последователен изход регистърът има и паралелни изходи (Q_1, Q_2, Q_3 и Q_4).

Представената на фиг. 12.3 схема може да се реализира и чрез JK тригери. Запазва се разгледаната структура, само че последователните входни сигнали постъпват на J входовете непосредствено, а на K входовете – след инверсия (следва от преобразуването на ЕА тип JK в D тип ЕА).

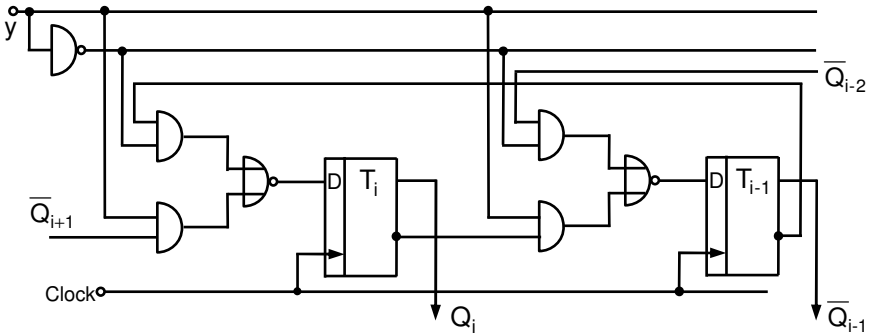
Аналогично може да се реализира преместване наляво, преместване на повече от един разряд и т.н.

Регистрите, които могат да работят в режим преместване наляво или надясно се наричат *реверсивни*, като пример за такъв регистър е схемата от фиг. 12.4.

Когато управляващият сигнал у е 1-ца, информацията се премества надясно, а когато у=0 – отдясно наляво. От анализа на тази схема се получава аналитичната зависимост (12.9)

$$D_i = \overline{y\overline{Q}_{i+1} + \overline{y}\overline{Q}_{i-1}}\,, \qquad (12.9)$$

така че при у=1 $D_i = Q_{i+1}$, а при у=0 $D_i = Q_{i-1}$.



Фиг. 12.4. Реверсивен регистър

Наличието на паралелни и последователни входове и изходи в преместващият регистър позволява те да се използват за преобразуване на кодови думи от паралелен в последователен код и обратно.

12.4. Регистри в интегрално изпълнение

Някои представители на регистри в интегрално изпълнение са: *SN7495* – 4 битов преместващ регистър с последователни и паралелни входове и изходи; *SN74LS164* – 8 разряден преместващ регистър; *SN7475* – 4 разряден паралелен регистър реализиран чрез D тригери. Въвеждането на входните сигнали x_i в регистъра става под действието на тактови сигнали (преход 0-1). Регистърът има два тактови входа (единият е за управление на 1-2 тригер, а вторият – за 3-4). При това не е необходимо предварителното нулиране на тригерите. Изведени са правите и инверсни изходи на всички тригери; *SN74100* – регистър състоящ се от осем D тригера, същото схемно решение както на *SN7475*. Има два тактови входа (1-4 и 5-8). Изведени са само правите изходи Q_i; *SN74175* - 4 разряден паралелен регистър, различаващ се от *SN7475* по това, че има асинхронен вход за едновременно нулиране на всички тригери; *SN74LS295* – тактовият вход Clock е един и изходите са с три състояния, което позволява той да работи с паралелни изходни линии, с други регистри и памети. Изходното състояние се управлява от входа x_z – при x_z=1 изходното ниво се определя от състоянието на тригерите на регистъра (регистърът се „чете"). При x_z=0 изходите са високоомни (намират се в състояние на висок импеданс).